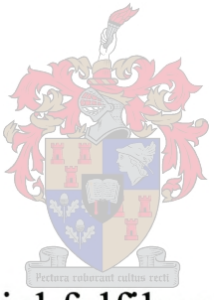
Simulation of the Switched Reluctance Machine under Single Pulse Mode Operation

Sibusiso W. Rasmeni



Thesis presented in partial fulfilment of the requirements for the degree of Masters of Science in Electrical Engineering Science at the University of Stellenbosch

Supervisor: Prof. M. J. Kamper

December 2003

Declaration

I, the undersigned, hereby declare that the work contained in this thesis is my own original work and that I have not previously in its entirety or in part submitted it at any university for a degree.

Abstract

Two distinct simulation methods have been developed to simulate the single and the multi-phase excited current waveforms of the switched reluctance machine (SRM) under single pulse mode operation. These simulation methods are explained and evaluated in this thesis. A non-commercial finite element package that takes the SRM's nonlinear magnetic property into account is used in this regard. The simulation program uses the finite element solution directly during the simulation. Both simulation methods have been investigated in terms of result and total simulation time. In this investigation it is shown how feasible the simulation methods will be with the next generation of fast computers. The factors affecting the simulated current waveforms of the SRM under single pulse mode operation are investigated in detail in this thesis. With these factors taken into account, the measured and simulated multi-phase current waveforms are compared with each other and discussed.

Opsomming

Twee verskillende simulasie metodes is ontwikkel om die enkel- en multi-fase stroomgolfvorms van die geskakelde reluktansiemasjien (GRM) onder enkelpulsmodus-werking te simuleer. In hierdie tesis word hierdie simulasie metodes verduidelik en geëvalueer. 'n Nie-kommersiële eindige element pakket wat die nie-linieêre magnetiese eienskap van die GRM in ag neem is in hierdie verband gebruik. Die simulasie program gebruik die eindige-element oplossing direk gedurende die simulasie. Beide simulasie metodes is ondersoek in terme van resultaat en simulasie-tyd. In hierdie ondersoek word getoon hoe uitvoerbaar hierdie tipe simulasie metodes gaan wees met die volgende hoë spoed generasie rekenaars. Die faktore wat die gesimuleerde stroomgolfvorms van die GRM onder enkelpulsmodus-werking beïnvloed word deeglik in hierdie tesis ondersoek. Met hierdie faktore in ag geneem word die gemete en gesimuleerde multi-fase stroomgolfvorms met mekaar vergelyk en bespreek.

Acknowledgements

I would like to express my sincere gratitude to:

- My promotor, Prof. M. J. Kamper, for his wonderful supervision and constant support and motivation, and also his unmeasurable efforts to ensure the availability of financial assistance for the project.

- Mr. P.D. Fick for his technical assistance and incredible advices.

- Dr. R. Wang , Mr. E. T. Rakgati and Mr. A. Dale for their consistent support and help during the project.

- All members of the machine group for their interest and support.

- The National Research Foundation (NRF) and the University of Stellenbosch for the financial support.

- My family and to all my friends including those in memory for their interest and encouragement.

- My son, Madoda, who had to grow up without me during the study period.

- The Almighty for giving me strength and endurance throughout my studies. Let the whole world praise his holy name for being unique and wonderful.

"To my grandmother in memory"

Contents

List of Figures

List of Tables

Glossary of Abbreviations

FEM	finite element model
SPMO	single pules mode operation
SRM	switched reluctance machine
DC	direct current
AC	alternating current
DSP	digital signal processor
RAM	Random access memory
PWM	pulse width modulation
I/O	input/output
EPLD	Erasable programmable logic device
ADC	analog-to-digital converter
DAC	digital-to-analog converter
IGBT	Isolated gate bipolar transistors

List of symbols

v_{cd}	Intantaneous value of the direct current voltage [V]
2 D, 3 D	Two and three dimensions
v_a, v_b, v_c	Instantaneous values of phases a, b and c voltages [V]
i_a, i_b, i_c	Instantaneous values of stator phases a, b and c currents [A]
λ_a, λ_b, λ_c	Instantaneous values of phase a, b and c flux-linkages [Wb]
L_a, L_c	Instantaneous values of phases a and c self-inductances [mH]
K_a, K_c	Constant values of phases a and c flux linkage change with position
M_{ac}, M_{ca}	Instantaneous values of phases a and c mutual-inductance [mH]
ω	Angular velocity [rad/s]
ω_m	Rotor angular velocity [rad/s]
T_l	Load torque [$N.m$]
T_m	Electromagnetic torque [$N.m$]
B	Total friction factor
T_{mc}, T_{ma}	Phases c and a electromagnetic torque [$N.m$]
J	Total moment of inertia [$kg.m^2$]
R_a, R_c	Phases a and c resistances [Ω]
Δi	Change in current [A]
$\Delta \theta$	Change in rotor position [$deg.$]
i_n, i_{n+1}	Stator phase currents at points n and $n+1$ [A]
ω_n, ω_{n+1}	Angular velocities at points n and $n+1$ [rad/s]
θ_n, θ_{n+1}	Rotor positions at points n and $n+1$ [$deg.$]
ξ, γ	Expressions for the known values from the previous calculations
Δt	Change in time [s]
ε	Is an element of
$\forall$	For all

Chapter 1

1 Introduction

Reluctance machines can be classified as either single-salient or double-salient reluctance machines. The single-salient reluctance machines have non-salient-pole stators but salient-pole rotors or rotors (non-salient or salient) with magnetic asymmetry. The other category of reluctance machine has saliency on both stator and rotor. Examples of this category are the stepping reluctance machines and switched reluctance machines. The focus of this dissertation is on the latter type of reluctance machines (SRMs). The machine under consideration has a number of concentrated windings mounted on the stator saliencies or "pole pieces". The rotor has no windings. Both the rotor and the stator are usually constructed from high permeability steel laminations stack together. As shown in Figure 1.1, the stator windings are energised sequentially; the excitation being switched from phase to phase so that the stator excitation stays just ahead of the rotor pole pieces as the rotor turns.

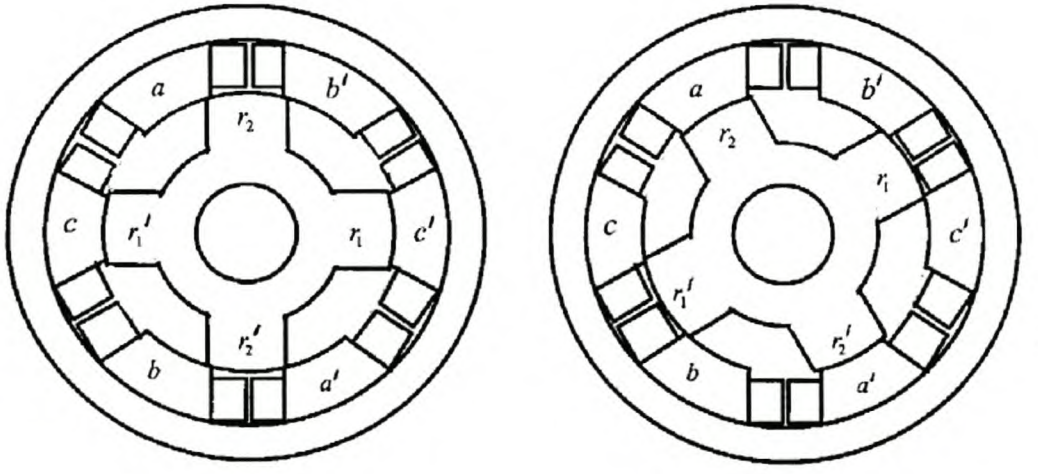


Figure 1.1 : SRM excitation of (a) phase a and (b) phase b

The main reason justifying the name 'Switched Reluctance Machine' is from the fact that the production of the torque and power at various rotor positions involves the switching of currents into the stator windings when there is a variation of reluctance.

Due to its construction, the SRM is robust and almost maintenance free. The advantages may be summarised as follows:

- The stator is simple to wind; the end turns are short and robust and have no phase-crossover.
- Although the flux pulsations in the rotor cause rotor iron losses, especially at high speed, most of the cooling of the machine is necessary on the stator, which is relatively easy to cool.
- The rotor is simple and easy to manufacture; it also has a low inertia.
- Because there are no magnets on the rotor the maximum permissible rotor temperature may be higher than in permanent magnet motors.
- The torque is independent of the polarity of phase current and for certain applications this permits a reduction in the number of power semi-conductor switches.
- Most converter circuits used for SRMs are resistant from shoot-through faults.
- If one phase is short circuited, the remaining phases can still work which implies that SRMs are fault tolerant.

Some of the disadvantages of the SRM drives are:

- the drives emit acoustical noise, need a converter, and information about the rotor position is necessary to have a proper control system.
- SRMs have non-linear magnetic circuits and a tendency to produce torque with high ripple. This is considered as making the SRM unsuitable for very low speed operation.

1.1 The history of the reluctance machine

The concept of rotating electrical machines was born out of the observation that an electromagnetic coil attracts a piece of iron. The force resulting from this attraction could then be used to obtain mechanical motion. The first reluctance machines were basically double-salient machines with electromagnets that were switched according to the position of moving iron in very much the same way as with the switched reluctance machine today [4]. The concept of the switched reluctance motor was established by 1838, but the motor could not realise its full potential until the modern

era of power electronics and computer-aided electromagnetic design [12]. Since the mid-1960s these developments have given the SR motor a fresh start and have raised its performance to levels competitive with d.c. and a.c. drives and brushless d.c. drives. During the last 15 years, some industrial applications have appeared and many efforts have now been made to improve the total performance of the switched reluctance drives. The major reasons for the use are the development of the microcontrollers and power electronic devices. In 1980, Professor Lawrenson was perhaps the first to adopt the term 'switched reluctance' in relation to the radial-airgap motor which is the focus of attention today, but the terms 'brushless reluctance motor', 'variable reluctance motor', and 'commutated reluctance motor' are among several equally acceptable alternatives that were in use long before this time [12]. The independent work by Lawrenson and Stephenson of Leeds University as well as Davis and Ray of Nottingham University brought about the potential these motors can offer in terms of efficiency and cost-effectiveness in comparison to other machines. They combined their development in 1973, and the first commercial outcome of their research was the Oulton variable speed drives.

Even to date there is still a growing interest in the Switched Reluctance Motors so that their application is being investigated in various fields of engineering, particularly in the field of traction because of their low weight/torque ratio and high reliability.

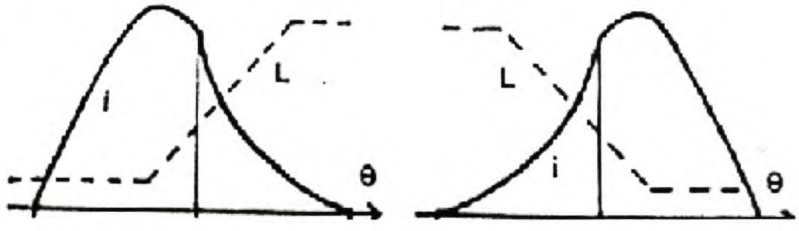
1.2 Problem statement

The SRM is a popular choice for high-speed applications primarily due to its robust rotor construction and its good high power density [5]. During high-speed operation, the SRM is said to operate in the so-called single pulse mode (SPM). In single pulse mode there is no current or voltage control of the SRM as the full DC link voltage is applied to the phase winding for the whole conduction period by means of the power electronic converter. It should be mentioned that under these conditions, the only required feed-back information for the machine drive is the position of the rotor.

As far as the author is concerned, no work has been published in literature that fully addresses the modelling of the SRM and the simulation of the SRM's single and

multi-phase current waveforms in SPM. As the main focus in this thesis, the SRM's single and multi-phase current waveforms together with the machine torque response under motoring SPMO at high speed are investigated.

In motoring single pulse mode operation (SPMO), when the rotor poles are approaching the alignment with the stator poles of the excited phase, the phase inductance increases from a minimum to a maximum as shown in Figure 1.2 (a). Motoring SPMO is attained by means of coinciding the pulses of the phase current with the period of positive rate of change of the excited phase inductance as again shown in Figure 1.2 (a). Generating (braking) SPMO is attained by coinciding the current pulse with a period of negative change of phase inductance as shown in Figure 1.2 (b). To prevent braking torque in motoring SPMO at high speeds, the phase winding must be de-energised at quite an advanced time (or rotor position) to allow the current to decay as shown in Figure 1.2 (a).



(a) (b)

Figure 1.2 : (a) High speed motoring SPMO and (b) high speed generating

1.3 Approach to problem

All problem aspects mentioned can be taken into account by using the FE solution actively during the simulation. The approach is thus to write a simulation program that calls, whenever it is necessary, a FE-program that solves for flux linkage and torque. A simple flow chart diagram illustrating this procedure is shown in Figure 1.3.

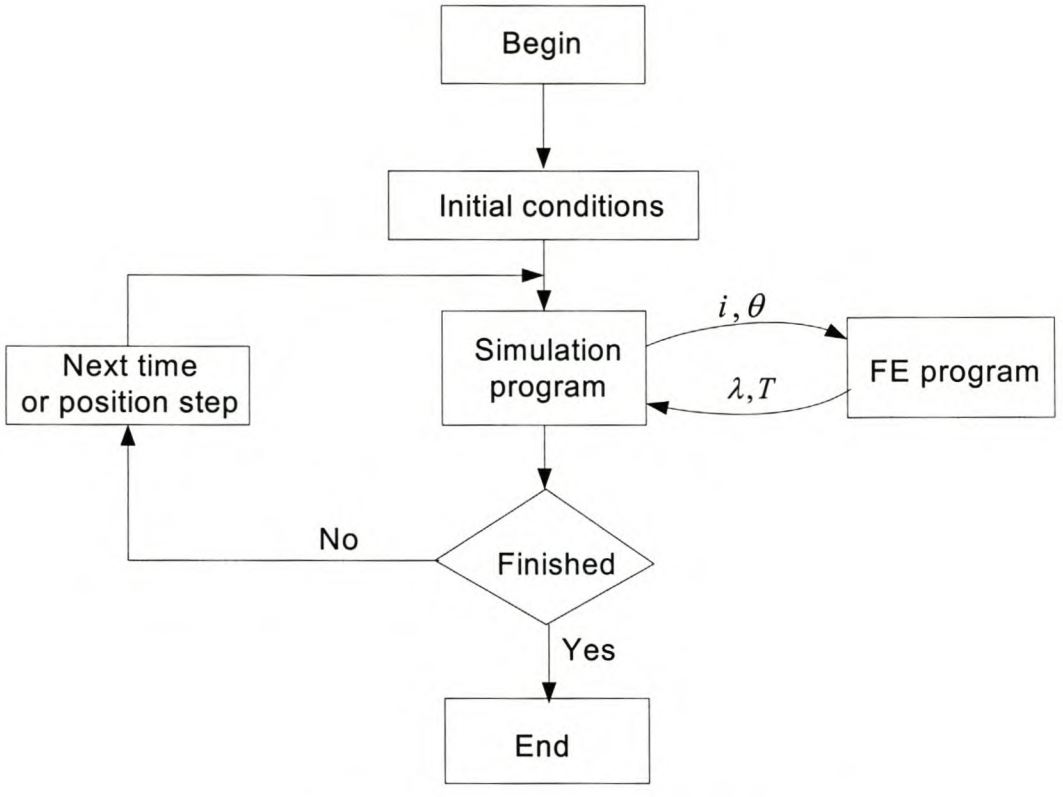


Figure 1.3 : Simulation procedure directly using the finite element solution

As will be discussed in Chapter 2, in the FE program (see Figure 1.3), the position of the rotor for each active stator phase is checked by considering graphically the FE field plots. This helps to decide at what position one should switch on and off a particular phase in the simulation in order to correctly simulate the motoring SPMO of the actual system. The simulation program (see Figure 1.3) solves for current from the dynamic voltage equations of the phases. The polynomial method of simulation together with the partial differential method of simulation are the two distinct simulation methods proposed in this thesis to do this.

As a unique feature of the switched reluctance motor, the phase windings are electrically separate from each other and have therefore small mutual coupling [1]. Taking advantage of this unique feature, the polynomial method of simulation is approached on the basis of one active stator phase winding only. In this method, at a particular rotor position the FE program is used to solve for the current value that has generated the known flux linkage value from the previous calculation. The FE program does this by using the interpolating polynomial technique of the Newton

form as given in detail in Chapter 2. The determined current value is, in turn, given to the FE program to get directly the motor torque value.

In contrast, the partial differential method is approached on the basis of multi-phase excitation, taking mutual coupling between the two adjacent phases into account. In this method, the phase currents and particular rotor position, are given to the FE program as inputs (see Figure 1.3). The generated torque value and total phase flux linkages are sent as outputs to the simulation program. Using these outputs, some of the parameters of the voltage equations, such as the self-inductances and mutual inductances are calculated in the simulation program. With the knowledge of these parameters and using the discrete format of Euler integration, new currents for the next step are solved-for as explained in Chapter 3. Furthermore, the new speed and position are also determined in the simulation program for the next time or position step, as also explained in Chapter 3. Again, a comparison is made between the measured and simulated current waveforms to determine the degree of correlation.

Finally, it should be mentioned that both simulation methods are investigated in terms of number of FE solutions and total simulation time in this thesis.

1.4 Thesis layout

The layout of the remainder of this thesis is as follows:

Chapter 2: The method for simulating the SR motor current waveform under SPMO is developed considering only one excited phase. The resulting simulated current waveform is finally compared with the measured current waveform to evaluate the degree of correlation.

Chapter 3: A second method using a different simulation algorithm is developed to simulate the SR motor current waveforms under SPMO. The approach is based on multi-phase excitation, though a single excited phase simulation is finally considered to investigate the effect of mutual coupling.

Chapter 4: Several factors affecting the current waveform under SPMO are considered. In this case only one phase is considered to be active and only the polynomial method described in Chapter 2 is used.

Chapter 5: The hardware of a digitally controlled SRM drive is described. Suitatable converter topology is given and discussed in more detail. The digital signal processor control unit used in the machine drive is also given and discussed.

Chapter 6: Measured single pulse mode multi-phase current waveform results are given and compared with the multi-phase simulated results to evaluate the degree of correlation.

Chapter 7: In this chapter a summary with conclusions are given and recommendations are made for further research work.

Chapter 2

2 Polynomial Method of Simulation

In this chapter, the main focus is at developing a mathematical model for simulating the current waveform of the tapered stator pole SRM under Single Pulse Mode Operation (SPMO). Only one phase excitation is considered in this simulation method. The method uses FE solution actively during the simulation, employing the method of interpolating polynomial of the Newton form to solve for the phase currents at each rotor position. In sections 2.1 to 2.4, some brief theoretical background information about the machine under consideration and the FEM package used is given. The last sections deal with the simulation condition and the simulation method used. Finally, a comparison of the measured with the simulated results is made.

2.1 Machine configuration

SRMs, are doubly salient machines with wound field coils for their stator windings and no coils or permanent magnets on their rotors as shown in Figures 2.1 and 2.2.

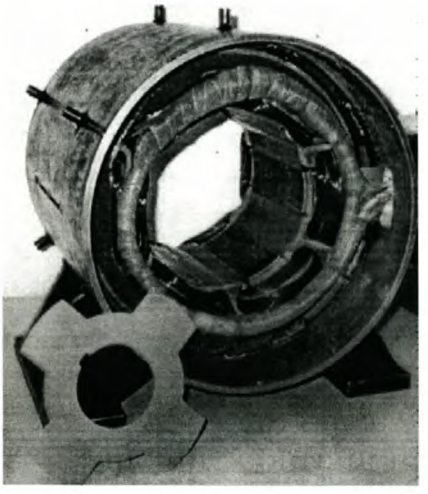


Figure 2.1: SRM active stator

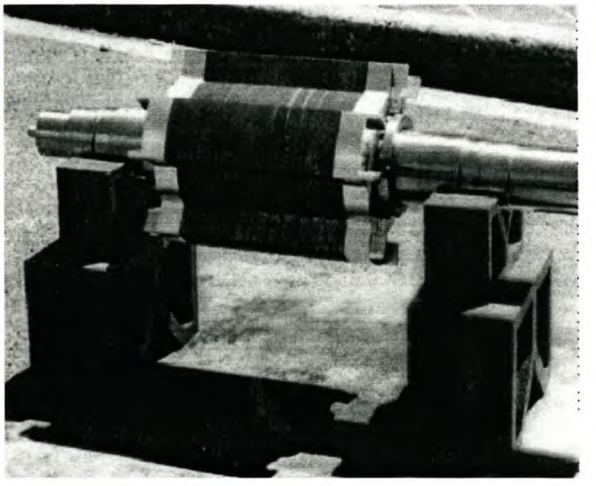


Figure 2.2: SRM passive rotor

There are however many design variations of SRMs, depending on their applications. The major differences encountered in the SRMs are in the pole shapes, stator-to-rotor pole ratios, number of teeth per pole and number of poles per phase. Preferably the stator to the rotor pole ratio should be a non-integer, even though some integer values have been attempted [1]. Most favoured configurations among many more options are the 6/4 three-phase (shown in Figures 2.1 and 2.2) and 8/6 four phase SRMs.

2.1.1 Pole shapes

One stator pole shape is dealt with in this thesis, i.e the tapered stator pole shape. With the tapered stator pole, the base and tip of the pole are not of the same width as shown in Figure 2.3. Furthermore, almost all the winding area of the stator slots are utilised. This implies that the tapered stator pole SRM has a good fill factor.

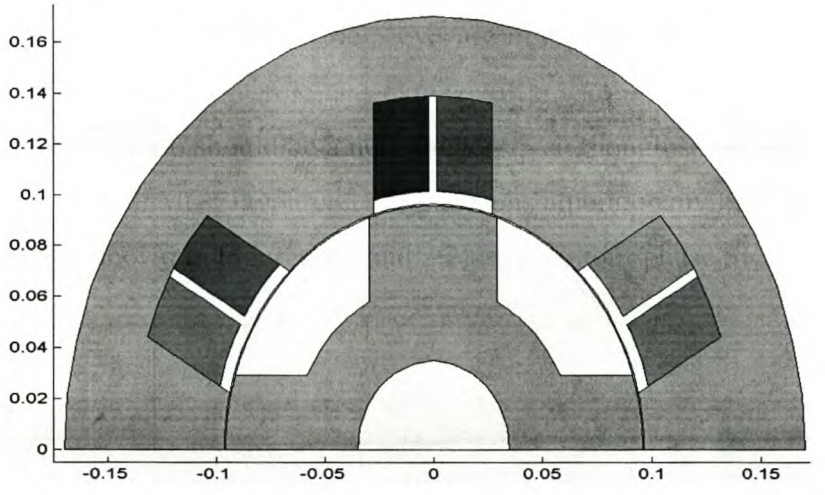


Figure 2.3: Tapered stator pole SRM

2.1.2 Number of phases

The choice of phase number is influenced in a major way by the required starting torque from any rotor position (and hence the effective values of $dL/d\theta$). To ensure adequate starting torque at all rotor positions there must be adequate overlap between the $L(\theta)$ variations of adjacent phases [5].

To provide a reversible SRM drive, that can develop positive and negative torques at any rotor position, a machine with at least three phases is required. Such a machine

could have six stator poles and four rotor poles, although other stator and rotor pole combinations are also possible as said before. In general, the more the number of poles and phases, the lesser the torque ripple problem is experienced in the machine. Increasing the number of poles and phases makes the excitation frequency of the stator phases higher for the same speed. In this thesis, only the three-phase 6:4 pole configuration tapered stator pole SRM shown in Figure 2.3, is considered as this machine is available from previous research studies [6] where the focus was on high speed traction applications.

2.2 FEM analysis

Accurate modelling is very important in the design stage of the SRM as it will be costly and time consuming to build and test the drive repeatedly. To avoid such costs and to serve time, several self-developed software programs have been used to model and analyse the SR motor. SRMs are highly magnetically non-linear. Due to this property, the software packages that take the non-linearity of the SRM's magnetic circuit are almost the only packages that are suitable for the accurate modelling of the SRM. FE analysis of a particular machine's magnetic structure can provide valuable information to formulate an accurate mathematical modelling of the machine.

It should be mentioned without any loss of trend; that the SRM cannot operate without a suitable converter and some sort of a controller. For this reason, drive designs and simulations must include these components. Later sections and chapters will show how this is done.

2.3 Rotor position confirmation

The unaligned position of the stator and the rotor poles for a particular phase means that the centre of the stator pole is between two rotor poles as shown in Figure 2.4. Before the simulation program was developed, it was necessary to know what angles in the FE program give the aligned and unaligned positions for a particular SRM phase. Using the FE package this was accomplished by considering field plots from

the FE solution as portrayed in Figures 2.4 and 2.5, where it is confirmed that 0° and 45° are the unaligned and aligned positions respectively. 15 *degrees* position is the position where the rotor and the stator pole start to overlap as shown in Figure 2.6. With this knowledge, a reasonable dwell angle (an angle between switch-on and switch-off positions) is chosen for each particular SR motor phase in the simulation program. The dwell angle is chosen in line with the principle of operation of the SR motor, which requires that each phase should be switched on earlier before the stator and the rotor pole overlap and be switched off when the conducting phase stator pole is on the verge of overlapping with the rotor pole.

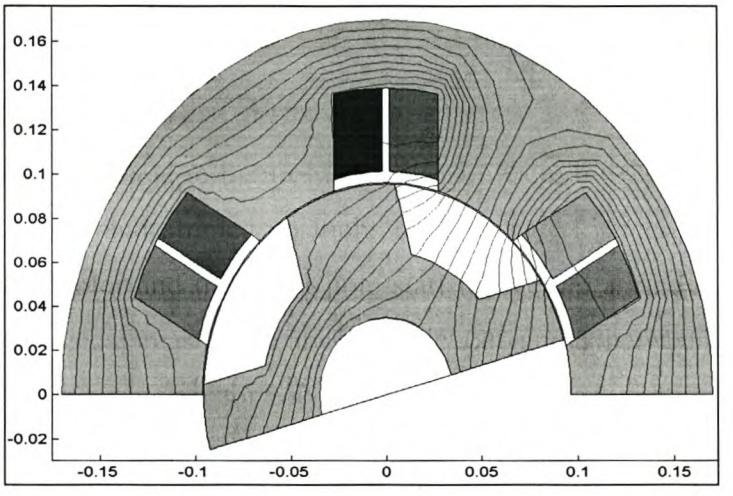


Figure 2.4: Finite element field solution of the tapered stator pole SRM at the-unaligned 0 degrees position

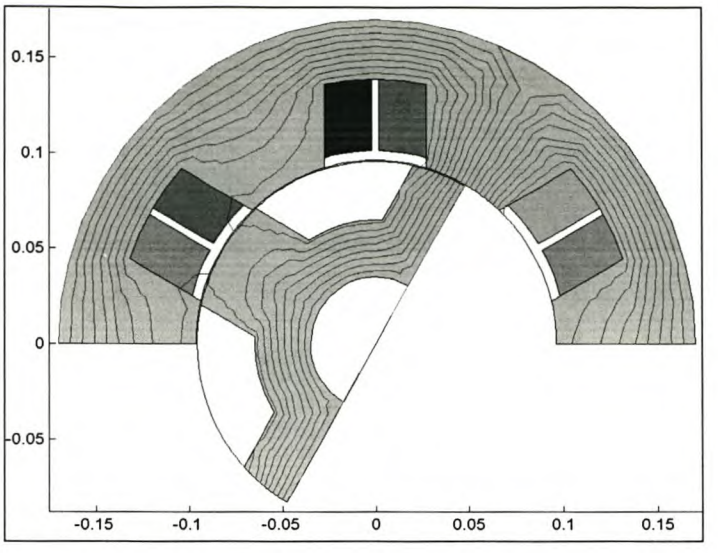


Figure 2.5 : Finite element field solution of the tapered stator pole SRM at the aligned 45 degrees position

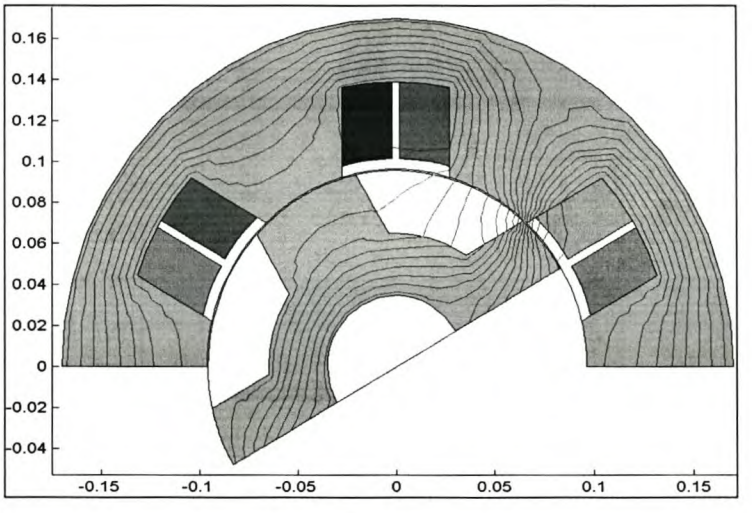


Figure 2.6 : Finite element field solution of the tapered stator pole SRM at 15 degree position

2.4 Non-linear modelling

Just like in conventional linear modelling, in non-linear modelling each phase of the SRM is considered as a coil with a constant resistance, R, and flux linkage, λ, which is dependent on the rotor angle, θ, and the phase current, i. If mutual coupling is ignored and it is assumed that no two phases are simultaneously active, each phase can be considered independently in the modelling. Thus, using Faraday's law, the terminal voltage for one phase is given by the relation:

$$v(\theta,i) = Ri + \frac{d\lambda(\theta,i)}{dt}. \qquad (2.1)$$

By means of the FE package used in this study, static FE solutions are used to determine λ as a function of θ and i. Table 2.1 portrays this flux linkage variation with the active stator phase current from unaligned-to-aligned positions. The outcome results are plotted in Figure 2.7. From these results it is seen that at unaligned positions, the flux-linkage variation with the phase current is linear. The reason for this behaviour is that, at an unaligned position the air gap is large. Therefore this is a region of maximum reluctance, which prevents the flux from saturating this particular region. As a result, unsaturated conditions are experienced. In the case of the aligned position, the flux linkage-current characteristic curve relationship is non-linear. The valid reason for this scenario is that the air gap is very small; hence the reluctance is approximately zero. As a result, the flux saturation is experienced which results in the non-linear flux linkage-to-phase current relationship.

Current	λ(0)	λ(5)	λ(10)	λ(15)	λ(20)	λ(25)	λ(30)	λ(35)	λ(40)	λ(45)
0	0	0	0	0	0	0	0	0	0	0
2	0.01	0.011	0.013	0.027	0.059	0.091	0.12	0.151	0.179	0.183
4	0.02	0.021	0.025	0.055	0.121	0.186	0.247	0.313	0.372	0.381
6	0.03	0.032	0.038	0.082	0.183	0.282	0.376	0.476	0.567	0.581
8	0.04	0.042	0.051	0.11	0.245	0.379	0.505	0.64	0.76	0.779
10	0.05	0.053	0.063	0.138	0.308	0.475	0.634	0.801	0.95	0.973
12	0.06	0.063	0.076	0.165	0.369	0.569	0.759	0.957	1.131	1.158
14	0.071	0.074	0.089	0.191	0.426	0.655	0.873	1.095	1.288	1.32
16	0.081	0.085	0.102	0.215	0.474	0.723	0.96	1.196	1.399	1.438
18	0.091	0.095	0.114	0.237	0.514	0.776	1.027	1.271	1.478	1.52
20	0.101	0.106	0.127	0.257	0.548	0.821	1.082	1.331	1.539	1.581
22	0.111	0.116	0.14	0.277	0.576	0.858	1.128	1.383	1.588	1.628
24	0.121	0.127	0.153	0.296	0.6	0.89	1.168	1.426	1.627	1.666
26	0.131	0.138	0.165	0.315	0.622	0.918	1.202	1.464	1.661	1.698
28	0.141	0.148	0.178	0.334	0.642	0.942	1.23	1.498	1.689	1.725
30	0.152	0.159	0.191	0.352	0.661	0.963	1.255	1.525	1.714	1.749
32	0.162	0.17	0.204	0.37	0.68	0.983	1.277	1.549	1.736	1.771
34	0.172	0.18	0.217	0.388	0.697	1.001	1.297	1.57	1.757	1.791
36	0.182	0.191	0.229	0.406	0.714	1.019	1.316	1.589	1.776	1.81
38	0.192	0.202	0.242	0.423	0.731	1.035	1.333	1.607	1.793	1.827
40	0.202	0.212	0.255	0.44	0.747	1.051	1.35	1.623	1.809	1.843
42	0.212	0.223	0.268	0.457	0.762	1.067	1.366	1.638	1.823	1.858
44	0.222	0.234	0.281	0.474	0.777	1.082	1.381	1.652	1.837	1.872
46	0.233	0.244	0.293	0.491	0.792	1.097	1.396	1.666	1.849	1.885
48	0.243	0.255	0.306	0.507	0.807	1.111	1.411	1.679	1.861	1.897
50	0.253	0.265	0.319	0.523	0.822	1.125	1.425	1.691	1.872	1.909
52	0.263	0.276	0.332	0.539	0.836	1.139	1.438	1.703	1.883	1.921
54	0.273	0.287	0.345	0.555	0.85	1.153	1.452	1.715	1.893	1.932
56	0.283	0.297	0.357	0.57	0.864	1.167	1.465	1.726	1.903	1.942
58	0.293	0.308	0.37	0.586	0.878	1.18	1.477	1.737	1.913	1.968

Table 2.1: Data for the flux linkage variation with current from unaligned to aligned positions

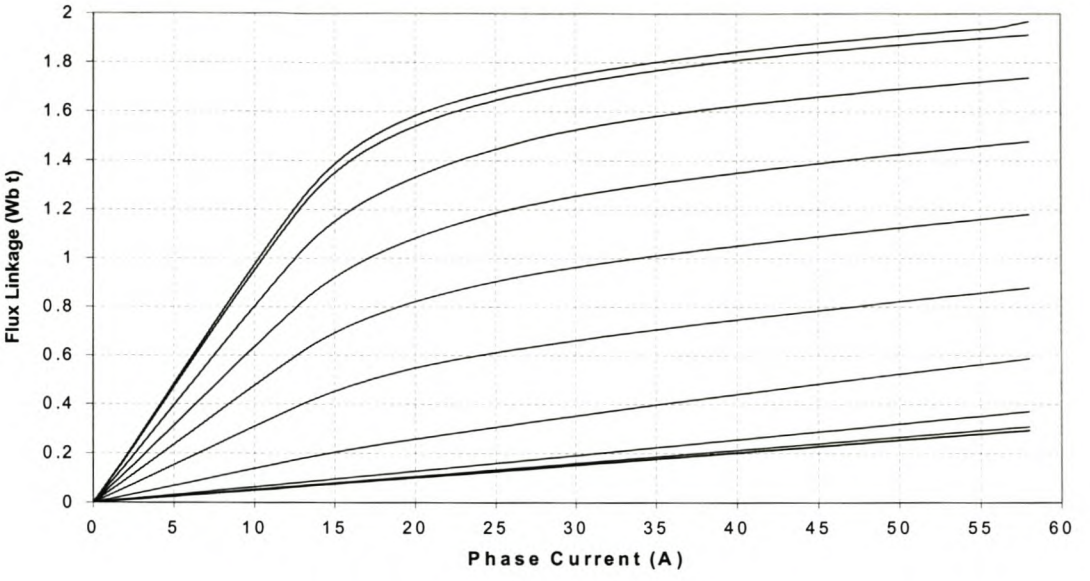


Figure 2.7: Flux linkage-current characteristic curves from unaligned-to-aligned positions

Furthermore, Figure 2.8 shows the static torque-angle profile of the tapered stator pole SRM with one phase active under rated current conditions of 110 *A*. It can be seen that this torque profile is not very smooth. This is due to some inadequate meshing of the stator and rotor poles [6]. The actual torque curve is very smooth. The solution to this problem is addressed in Chapter 4.

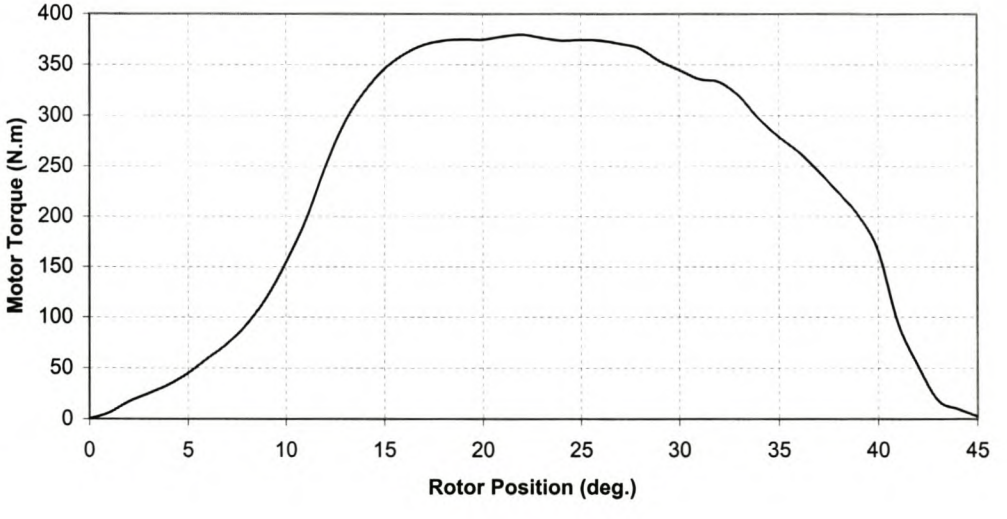


Figure 2.8: Phase-torque versus rotor position at a constant phase-current of 110 A

2.5 Single pulse mode operation of the SRM

At high speeds the SRM drive system operates in the so-called single pulse mode. This means that the full DC bus voltage is applied to the phase windings of the machine for the whole switched-on angle by means of a power electronic converter, whose circuit diagram is given in Figure 2.9. The half-bridge phase leg circuit can supply current in only one direction. Since motoring SPMO is dealt with in this thesis, only a positive voltage is applied on the phase windings of the machine, as a result of the current regulation strategy used. The regulation strategy suitable for operating the SRM as a motor is discussed in more detail in Chapter 5.

As a result of employing this converter topology, the current through the phase winding is no longer under control in SPMO and is determined by the bus voltage, flux linkage, rotor position and the speed of the machine. Also, the generated torque is no more under control and is dependent on the current through the phase winding. It is not simple to accurately determine the current and the torque of the machine drive in the single pulse mode as the magnetic circuit varies and is non-linear due to saturation. Furthermore, the pulsating torque might cause speed variations that affect the induced voltage and hence the current. The latter obviously depends very much on the inertia of the whole drive system.

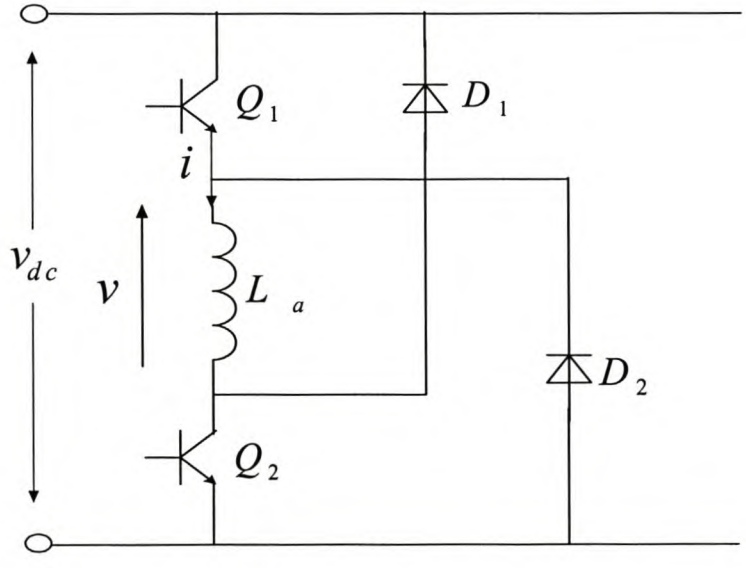


Figure 2.9: Half-bridge phase leg for SRM's operation

2.6 Conventional simulation method

To solve for the current and torque of the SRM the conventional method is firstly to obtain a complete set of flux linkage data of the machine through measurements or finite element analysis. This data is then used to solve, by means of interpolating polynomials (curve fitting), look-up tables and numerical methods the current in the phase winding. An example in literature where this is done is in [10]. Also from the flux linkage data the torque is calculated. This is a less simple procedure and it becomes even more difficult if two phases are active, which is typically the case in the high-speed single pulse mode where the phase currents are overlapping. To take overlapping of currents into account receives seldom attention in literature. The advantage of the above method is that when curve fitting and other derivations and pre-processing have been finished, the current and the torque response can be simulated quite fast for basically all operating conditions of the machine. The disadvantage of this method, however, is that, when different designs are investigated the processing of the flux linkage data for all different designs have to be done.

2.6 Finite element simulation technique

A proposal and evaluation of a stepping procedure whereby FE analysis is directly used in the simulation of the current and torque of the SRM drive in the single pulse mode is dealt with in this section. In the simulation of the current and torque response the following well-known equations are solved:

$$v = Ri + \frac{d\lambda}{dt} \tag{2.7}$$

$$T_m - T_l = J\frac{d\omega}{dt} \tag{2.8}$$

$$\frac{d\theta}{dt} = \omega \tag{2.9}$$

A simulation block diagram of these equations is shown in Figure 2.10. Note that in this case only one phase is active in the simulation. It is thus assumed that overlapping of the phase currents does not occur. To simulate the current and the speed variations in single pulse mode, a DC voltage equal to the DC link voltage of the converter is given as an input to the block diagram of Figure 2.10. The outputs are current and speed. At switch-off, a negative DC voltage ($-v_{dc}$) is applied to the phase winding. The phase winding demagnetises through the two freewheel diodes of Figure 2.9. The very heart of the simulation is the two functional blocks A and B of Figure 2.10 where flux linkage is converted to current and current to torque respectively, using finite-element analysis.

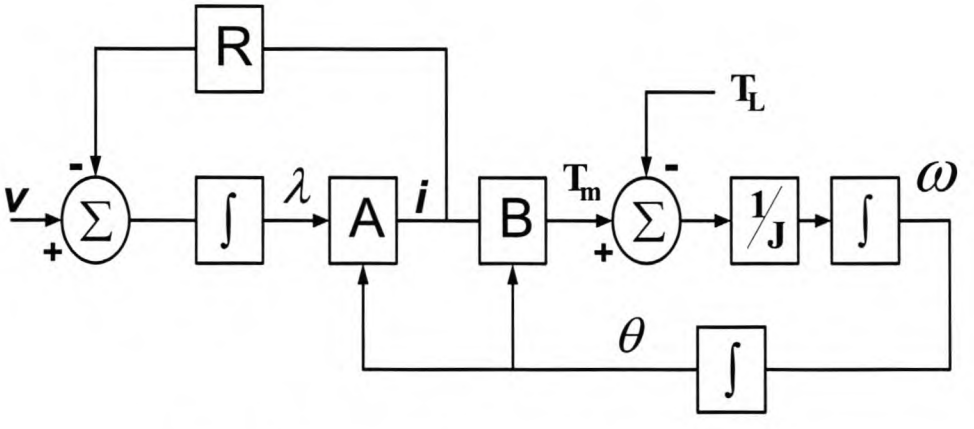


Figure 2.10: Simulation block diagram (one phase active).

The inputs to functional block A of Figure 2.10, to determine the current, are the latest values of the flux linkage, λ, and the rotor position, θ, of the machine. With these inputs known, the FE analysis program is called to determine the current, *i*, that generates λ at the rotor position θ. To explain this, consider the flux-linkage-versus-current curve of Figure 2.11. The flux linkage, λ, (on the y-axis) is known from previous simulation calculation. The objective now is to find the current value, *i*, (shown on the x-axis) that generates λ. This will then be the solution of functional block A. To determine this current, the flux linkage, λ, is first bracketed. To bracket the flux linkage, λ, three current values, i_1, i_2 and i_3 are determined and given to the FE-program to calculate three corresponding flux linkage values λ_1, λ_2 and λ_3 at the

rotor position, θ. To accomplish such bracketing, the basic requirement is that, on one hand, if $\lambda_1 < \lambda_2 < \lambda_3$, then $\lambda_1 < \lambda < \lambda_3$, on the other hand, if $\lambda_3 < \lambda_2 < \lambda_1$, then $\lambda_3 < \lambda < \lambda_1$, depending on the nature of the initially determined current values, i_1, i_2 and i_3. The three i-values that bracket λ, as shown in Figure 2.11, are used to do a curve fitting using a second degree interpolating polynomial of the Newton form. The variation of flux linkage with current gives a smooth curve so that the use of just three points for curve fitting is justified. Hence, with the second-degree polynomial of the Newton form, λ as a function of i can be written as

$$\begin{aligned}\lambda(i) &= c_1 + c_2(i - i_1) + c_3(i - i_1)(i - i_2) \\ &= c_3 i^2 + [c_2 - (i_1 + i_2)c_3]i + (c_1 - c_2 i_1 + c_3 i_1 i_2)\end{aligned} \tag{2.10}$$

where c_1, c_2 and c_3 are the polynomial constants, given by the following relations:

$$c_1 = \lambda(i_1) \tag{2.11}$$

$$c_2 = \frac{\lambda(i_1) - \lambda(i_2)}{(i_1 - i_2)} \tag{2.12}$$

$$c_3 = \frac{c_2}{(i_1 - i_3)} - \frac{\lambda(i_1) - \lambda(i_3)}{(i_2 - i_3)(i_1 - i_3)}. \tag{2.13}$$

Knowing these constants and $\lambda(i)$, the current, i, can be solved from (2.10) using the solution of a quadratic equation, given by the relation

$$i = \frac{2ac}{-(b \pm \sqrt{b^2 - 4ac})}, \tag{2.14}$$

where $a = c_3$, $b = c_2 - c_3(i_1 + i_2)$, $c = -\lambda(i) + c_1 - c_2 i_1 - c_3 i_1 i_2$

Solving for the current from (2.10) will require at least three FE solutions. With the current and the position known as inputs to functional block B of Figure 2.10, the torque can be calculated through another FE solution. Thus, to solve functional blocks A and B of Figure. 2.10 will take four FE solutions. Note, however, that the torque

can also be calculated from the first three FE solutions through curve fitting. This will save one FE solution. Furthermore, if the bracketing is close, then linear interpolation through two points can be used, which will save another FE solution. Thus, a minimum of two and a maximum of four FE field solutions are required to solve the functional blocks A and B of Figure 2.10. Finally, it is important to note that the solved reluctivities of the previous non-linear FE field solution are used in the next field solution. This saves a lot of simulation time as the previous reluctivity values are already close to the next (new) reluctivity values.

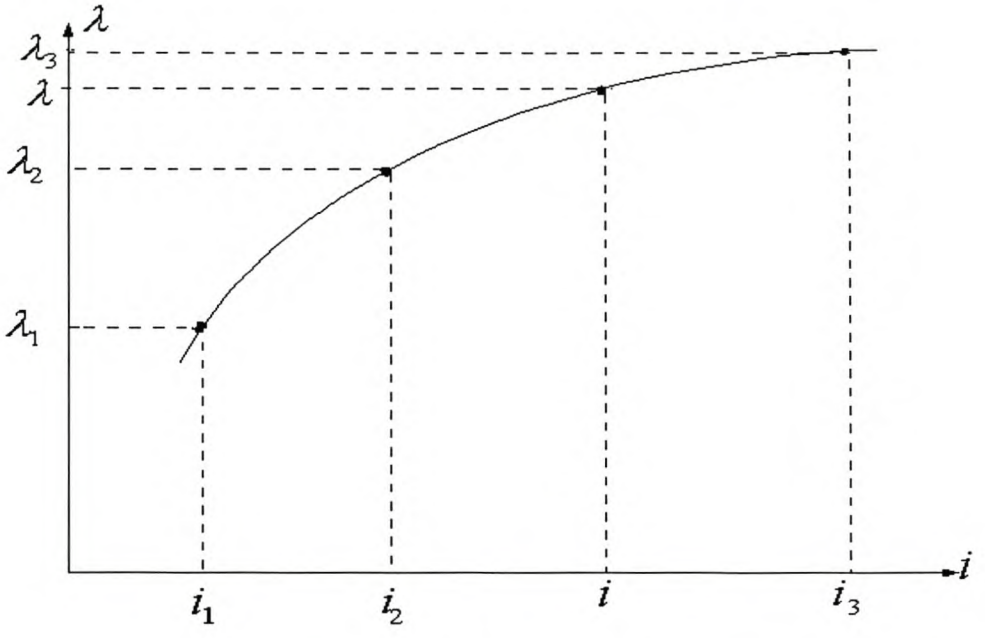


Figure 2.11: Curve fitting through three points.

With A and B solved, the rest of the simulation can easily be done according to Figure 2.11. For the integration, Euler integration is used. The code for the program is given in Appendix B.

2.7 Simulated and measured results

As a first step in the investigation of the single pulse current waveform, the speed of the machine is taken as constant in the simulation, with the assumption that the inertia of the system is very large. The system is set at 1500 r/min and a DC bus voltage of

500 V is used. The SRM used in the simulation is the 50 kW machine described earlier in this chapter. The result of the direct FE simulated single pulse current waveform is shown in Figure 2.12.

Also shown is the measured current waveform result of the SRM. Although the comparison is truly speaking not valid, because the actual inertia is not infinite and the actual phase currents are overlapping, the measured and simulated results show to some extent an agreement. It should be taken into consideration that 2D FE analysis is used and that actual 3D effects are thus ignored.

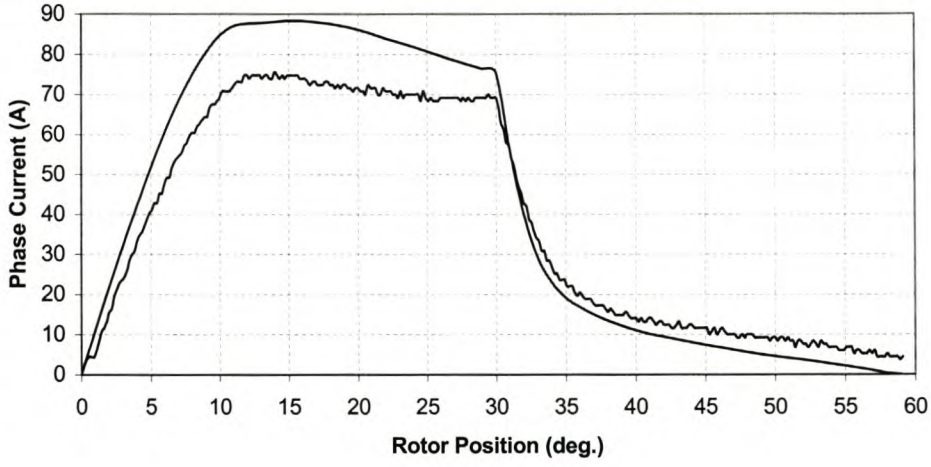


Figure 2.12: Direct finite element simulated (smooth line) and measured (rough) current waveforms

It can be seen that the simulated peak current is higher than the measured one. The first possible reason for this discrepancy might be a higher inductance of the actual machine system compared to low inductance level of the 2D simulation system. The other possible reason might be due to the dwell angle delay in the simulation results. In Chapter 4, however, it will be shown how drastically the dwell angle delay affects the SRM current waveform under SPMO.

2.8 Simulation time evaluation

In this section, an attempt to show how the simulation program using the polynomial method of simulation improves in terms of the simulation time with the old and the

new computers is made. An 800 *MHz and* a 1.67 *GHz* computer are used to perform such evaluation. Tables 6.1 and 6.2 show the number of field solutions, time per field solution and the total simulation time taken using the two computers.

Computer speed (*MHz*)	Time step (μs)	Time per field solution (*s*)	Number of field solutions	Total simulation time (*min*)
800	110	18	60	18

Table 2.2: Information about simulation results

Computer speed (*GHz*)	Time step (μs)	Time per field solution (*s*)	Number of field solutions	Total simulation time (*min*)
1.67	110	8	60	8

Table 2.3: Information about simulation results

From Tables 2.2 and 2.3 it is noticed that an improvement from the 800 *MHz* to 1.67 *GHz* computer has reduced the simulation time by a reduction factor of more than *2-times* in magnitude. Due to the faster simulation process observed with the use of the new computer, it can be concluded that for 5-10 *years* to come, the same simulation program will take even less than a minute to complete.

Chapter 3

3 Partial Differential Method of Simulation

In this chapter another method for the current waveform simulation, called partial differential method, is described. Using this method, the current waveform simulation is approached on the basis of multi-phase simulation (two phase currents overlap is taken into account) in a single pulse mode operation. Hence, a multivariable situation is considered in this regard. Simulation steps to find phase current solutions at any rotor position using FE solutions are discussed in more detail. Comparison of a single-phase simulated current waveform under multi-phase conditions with the simulated current waveform under single-phase excited conditions is done to evaluate the degree of the effect of mutual coupling. Finally, the single-phase current waveform results of the partial differential simulation method are compared with the single-phase current waveform results of the polynomial simulation method, to evaluate the correlation between these simulation results.

3.1 SRM electrical and mechanical equations

To model the actual SRM's SPMO, two adjacent phases conducting at a time should be taken into account, where one phase is the outgoing (switch-off) phase while the other phase is the incoming (switch-on) phase. Outgoing phase c and incoming phase a are considered as the SRM's active phases in this section. The electrical and mechanical equations used in the simulation are also derived and explained. The equivalent circuits of phases c and a are also given according to their derived electrical equations.

3.1.1 Electrical equations

The terminal voltage equation governing the flow of stator current may be written as

$$v = Ri + \frac{d\lambda}{dt} \tag{3.1}$$

with $$v = \begin{bmatrix} v_c \\ v_a \end{bmatrix}, \quad i = \begin{bmatrix} i_c \\ i_a \end{bmatrix} \text{ and } \quad \lambda = \begin{bmatrix} \lambda_c \\ \lambda_a \end{bmatrix}, \tag{3.2}$$

where v is the voltage (of appropriate polarity) applied across the winding, i is the phase current and λ is the flux linking the coil. Since two active phases are considered at a time, which are phases c and a in this case, their flux linkages are both functions of the current in phase c, i_c, the current in phase a, i_a, and the rotor position, θ, thus

$$\lambda_c = f(i_c, i_a, \theta) \quad \text{and} \quad \lambda_a = f(i_c, i_c, \theta)$$

Hence, from equation (3.1), it follows that:

$$v_c = Ri_c + \frac{d\lambda_c(i_c, i_a, \theta)}{dt} \tag{3.3}$$

and $$v_a = Ri_a + \frac{d\lambda_a(i_c, i_a, \theta)}{dt}. \tag{3.4}$$

Since i_a, i_c and θ, are in turn functions of time, t, it follows from the chain rule mathematical theorem given in Appendix A that:

$$\begin{aligned} \frac{d\lambda_c}{dt} &= \frac{\partial\lambda_c}{\partial i_c}\frac{di_c}{dt} + \frac{\partial\lambda_c}{\partial i_a}\frac{di_a}{dt} + \frac{\partial\lambda_c}{\partial\theta}\frac{d\theta}{dt} \\ &= L_c\frac{di_c}{dt} + M_{ca}\frac{di_a}{dt} + K_c\omega \end{aligned} \tag{3.5}$$

and $$\frac{d\lambda_a}{dt} = \frac{\partial\lambda_a}{\partial i_a}\frac{di_a}{dt} + \frac{\partial\lambda_a}{\partial i_c}\frac{di_c}{dt} + \frac{\partial\lambda_a}{\partial\theta}\frac{d\theta}{dt}$$

(3.6)

$$= L_a \frac{di_a}{dt} + M_{ac} \frac{di_c}{dt} + K_a \omega$$

From equations (3.5) and (3.6) together with equations (3.3) and (3.4) the phase c and the phase a voltage equations of the SRM become:

$$v_c = Ri_c + L_c \frac{di_c}{dt} + M_{ca} \frac{di_a}{dt} + K_c \omega \tag{3.7}$$

and

$$v_a = Ri_a + L_a \frac{di_a}{dt} + M_{ac} \frac{di_c}{dt} + K_a \omega \ . \tag{3.8}$$

Equations (3.7) and (3.8) can be visualised through the circuits of Figure 3.1, which are the SRM phase equivalent circuits. According to [9], unlike most machines, these equivalent circuits are not very helpful, since the "constants" L_c, L_a, K_c and K_a vary with θ, i_c and i_a.

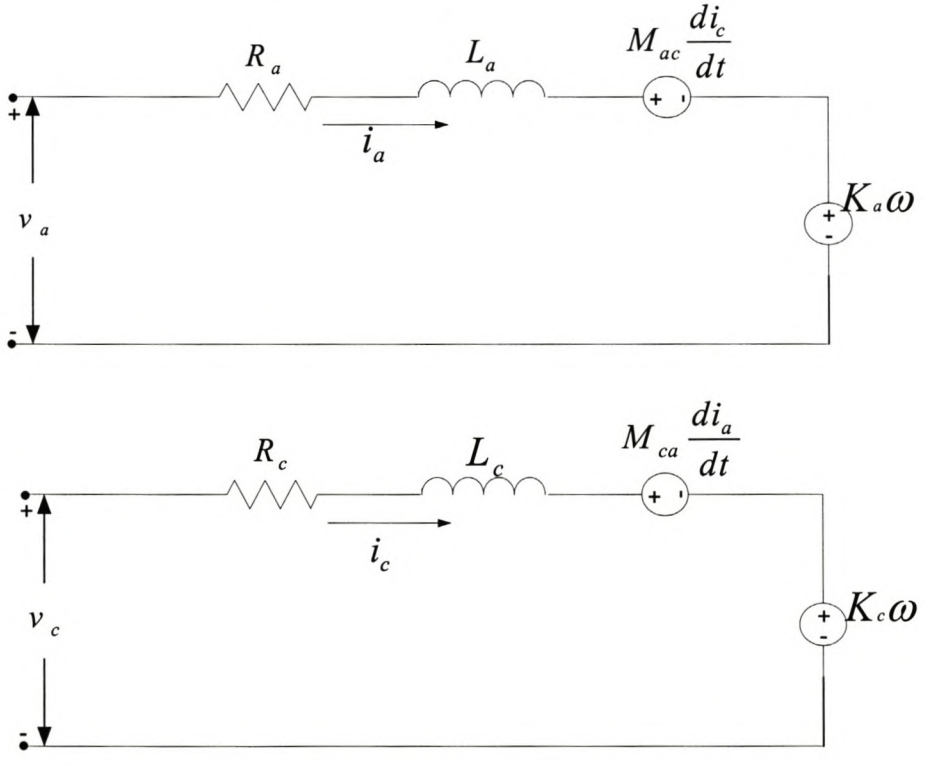


Figure 3.1: The SRM phase equivalent circuits

3.1.2 Mechanical equation

The mechanical equation for the system is given by the relation

$$T_m(i_c, i_a, \theta) - T_l = J\frac{d\omega_m}{dt} + B\omega_m, \tag{3.9}$$

where T_m is the motor torque, J is the motor inertia, B is the friction coefficient, and T_l is the load torque (carefully chosen as a constant value). During the multi-phase operation discussed in this chapter, the torque request is distributed between the two excited phases during the commutation interval, and can be expressed according to the mathematical relation

$$T_m(i_c, i_a, \theta) = T_{mc}(i_c, \theta) + T_{ma}(i_a, \theta). \tag{3.10}$$

The electrical equations as well as the mechanical equations are expressed in a block diagram format as shown in Figure 3.2.

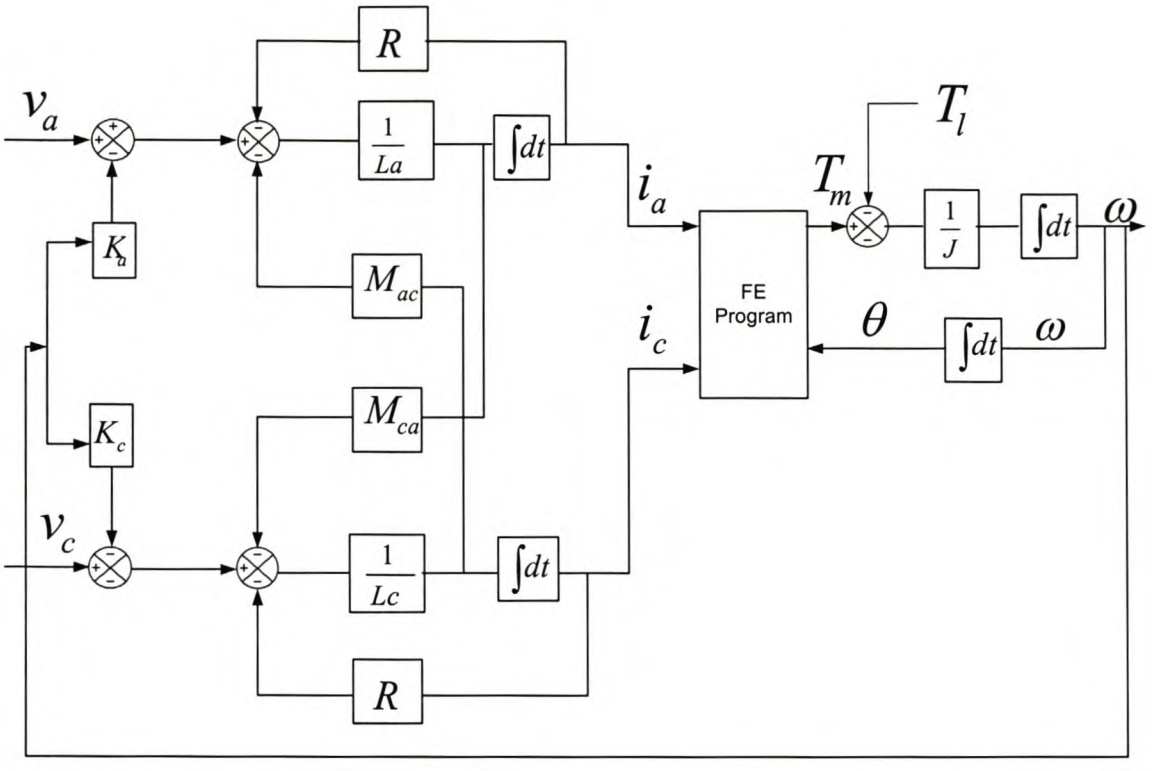


Figure 3.2: Simulation block diagram

3.2 FE simulation

In this section the partial differential method used to simulate multi-phase SRM's current waveform using FE solutions under SPMO is discussed in more detail. This simulation method uses both the electrical and the mechanical SRM system's equations as portrayed in Figure 3.2. This implies that, unlike in the previous simulation method, the actual system's moment of inertia is used in this simulation method, which is a finite value. The actual system's factor that is not taken into account in the simulations is the effect of the core losses. The friction coefficient, B, in equation (3.9) is also assumed to be extremely small and is ignored.

3.2.1 Simulation Method

In the simulation of the current and the torque response, equations (3.7), (3.8) and (3.9) are solved. The simulation block diagram of these equations is shown in Figure

3.2. From equations (3.7) and (3.8), the phase currents for the adjacent active phases, i_c and i_a, at a particular rotor position, θ, are known from the previous calculation. The DC bus voltage for the active phases is also known. As shown in the block diagram, for each rotor position, θ, and with the input phase currents given to the FE program, the corresponding generated motor torque, T_m, is obtained as the output. With the knowledge of these parameters determined from the previous calculation, the self-inductances, L_c and L_a, the mutual inductances, M_{ac} and M_{ca}, as well as the constants K_c and K_a are determined by means of the FE program. The phase inductances to be determined are the instantaneous inductances. Therefore, for clarity purposes, the manner in which the calculations are carried out in the simulation program is given with the aid of the λ versus i SRM's characteristic curves shown in Figures 3.3a and 3.3b. As shown in the figures, at a particular rotor angle, θ, each of the active phase currents, i_c and i_a, is changed by a $\pm \Delta i$ constant value. The resulting current values, $i_c - \Delta i$, $i_c + \Delta i$, $i_a - \Delta i$ and $i_a + \Delta i$, are given to the FE program, which, calculates by means of FE solutions, the respective corresponding self-flux linkages, λ_c, λ'_c, λ_a and λ'_a.

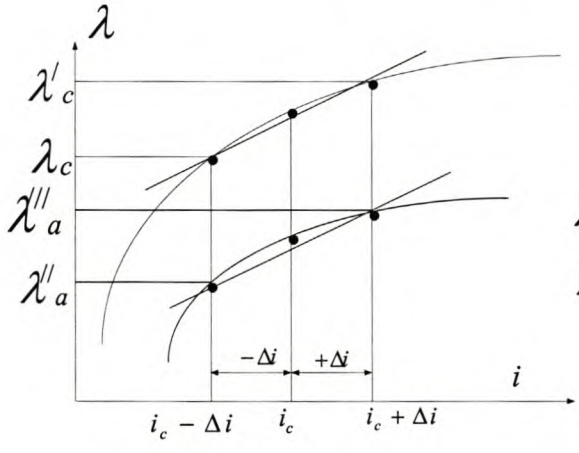


Figure 3.3a

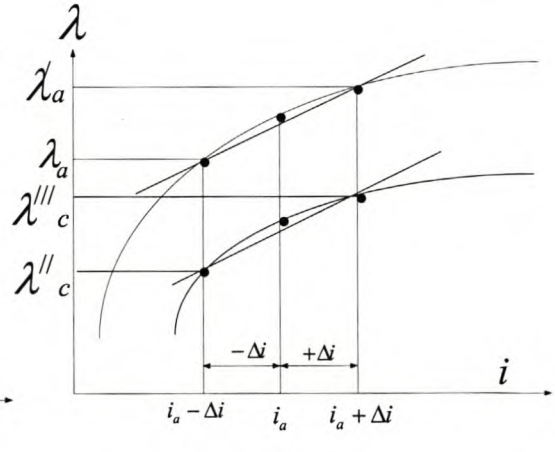


Figure 3.3b

With this information available in the simulation program, the self inductances, L_c and L_a are calculated from the following mathematical relations:

$$L_a = \frac{\lambda'_a - \lambda_a}{(i_a + \Delta i) - (i_a - \Delta i)} = \frac{\lambda_a{}' - \lambda_a}{2\Delta i} \tag{3.11}$$

and

$$L_c = \frac{\lambda'_c - \lambda_c}{(i_c + \Delta i) - (i_c - \Delta i)} = \frac{\lambda_c{}' - \lambda_c}{2\Delta i}. \tag{3.12}$$

Due to mutual coupling between the two adjacent active phases, each of the active phase currents will have an effect on its adjacent phase winding. Therefore, mutual flux linkages between these adjacent phases will be attained. This implies that the current values, $i_c - \Delta i$, $i_c + \Delta i$, $i_a - \Delta i$ and $i_a + \Delta i$, apart from the self flux linkages, λ_c, λ'_c, λ_a and λ'_a, will also result to the respective corresponding mutual flux linkages, λ''_a, λ'''_a, λ''_c and λ'''_c to be calculated by the FE solution as shown in the Figures 3.3a and 3.3b. Therefore, mutual inductances, M_{ac} and M_{ca}, can be calculated in the simulation program, given by the relations

$$M_{ac} = \frac{\lambda'''_a - \lambda''_a}{(i_c + \Delta i) - (i_c - \Delta i)} = \frac{\lambda_a{}''' - \lambda''_a}{2\Delta i} \tag{3.13}$$

and

$$M_{ca} = \frac{\lambda'''_c - \lambda''_c}{(i_a + \Delta i) - (i_a - \Delta i)} = \frac{\lambda_c{}''' - \lambda''_c}{2\Delta i}. \tag{3.14}$$

So far at this point, four FE solutions have been found. Furthermore, employing the same principle, the rotor position, θ, is changed by $\pm\Delta\theta$ constant value. Bearing in mind that two phases are active at each of the positions, $\theta - \Delta\theta$ and $\theta + \Delta\theta$, it follows that at each of the position values, two corresponding phase current values simultaneously given to the FE program result in the corresponding flux linkage values being obtained in the simulation program. This implies that at the position values, $\theta - \Delta\theta$ and $\theta + \Delta\theta$, corresponding flux linkage values λ°_c, λ°_a and λ^*_c, λ^*_a will be obtained respectively. Hence, the constants, K_a and K_c, are determined in the simulation program using the mathematical relations

$$K_a = \frac{\lambda^*_a - \lambda^\circ_a}{(\theta + \Delta\theta) - (\theta - \Delta\theta)} = \frac{\lambda_a^* - \lambda^\circ_a}{2\Delta\theta} \tag{3.15}$$

and

$$K_c = \frac{\lambda^*_c - \lambda^\circ_c}{(\theta + \Delta\theta) - (\theta - \Delta\theta)} = \frac{\lambda_c^* - \lambda^\circ_c}{2\Delta\theta}. \tag{3.16}$$

The constants, K_a and K_c, are thus determined from two FE solutions. This implies that a total number of six FE solutions are required per time or position-step for the partial differential simulation technique.

With the knowledge of all these parameters, the new phase currents can be solved-for. Since a time-step solution of the currents of Figure 3.2 is done using Euler's integration, therefore, the derivative of the current as a function of time is determined by approximation as

$$\frac{di}{dt} \approx \frac{i_{(n+1)} - i_{(n)}}{\Delta t}, \tag{3.17}$$

where $i_{(n)}$ is the known current value at time-step n and $i_{(n+1)}$ is the new unknown current value at time-step $n+1$. Substituting equation (3.17) in equations (3.7) and (3.8), results to the following system of linear equations:

$$L_a i_{an+1} + M_{ac} i_{cn+1} = \xi, \text{ with} \tag{3.18}$$

$$\xi = V_a \Delta t - i_{a_n} R\Delta t + L_a i_{an} + M_{ac} i_{cn} - K_a \omega_n, \tag{3.19}$$

and

$$M_{ca} i_{an+1} + L_c i_{cn+1} = \gamma \tag{3.20}$$

$$\gamma = V_c \Delta t - i_{c_n} R\Delta t + L_c i_{cn} + M_{ca} i_{an} - K_c \omega_n, \tag{3.21}$$

where i_{cn} and i_{an} are the current values from the previous calculations, i_{cn+1} and i_{an+1} are the new current values to be determined, ω_n is the known speed value from the previous calculation, and γ and ξ are the expressions of the known values determined from the previous calculation. Equations (3.18) and (3.20) are then solved

simultaneously to get the new current values in the simulation program. With the knowledge of the new currents, the new motor torque is determined by means of the FE program. Assuming that the friction coefficient in equation (3.9) is extremely small (approximately equal to zero), the equation reduces and can be expressed in the form

$$\frac{d\omega}{dt} = \frac{1}{J}(T_m - T_l) . \tag{3.22}$$

Again, since a time-step solution of the speed of Figure 3.2 is done using Euler integration, the speed derivative is determined by approximation as

$$\frac{d\omega}{dt} \approx \frac{\omega_{(n+1)} - \omega_{(n)}}{\Delta t}, \tag{3.23}$$

where $\omega_{(n+1)}$ is the new speed. Substituting equation (3.23) in equation (3.22), results in equation (3.22) being resolved into the form

$$\omega_{(n+1)} = \frac{1}{J}(T_m - T_l)\Delta t + \omega_n . \tag{3.24}$$

In the simulation program the new speed is calculated according to equation (3.24). The knowledge of the new motor speed allows the determination of the new position from the relation

$$\theta_{(n+1)} = \omega_{(n+1)}\Delta t + \theta_{(n)} , \tag{3.25}$$

where $\theta_{(n)}$ and $\theta_{(n+1)}$ are the old and the new positions, respectively. Finally, all the new parameters determined are saved as old values to be ready for the next time or position step.

3.3 Simulation results

In this section a speed profile showing extremely small speed variation with the rotor position during the simulations is given. The simulated multi-phase current waveforms are also given and briefly discussed. Resulting multi-phase torque profile

is also dealt with. Finally, the self-inductance and the mutual inductance profiles during the SPMO are also given.

3.3.1 Simulation conditions

As a first step in the current waveform simulation under SPMO using the partial differential method of simulation, the system's initial speed value is set at 1500 rpm and a DC bus voltage of 500 V is used. The moment of inertia used is the actual system's moment of inertia, which is equal to 0.8 kg.m^2. The load torque is taken as constant. From the discrete speed equation (3.24), due to the motor torque variation from one position step to another, the rotor speed also varies. The simulated speed profile of Figure 3.4 shows that such a variations is very small.

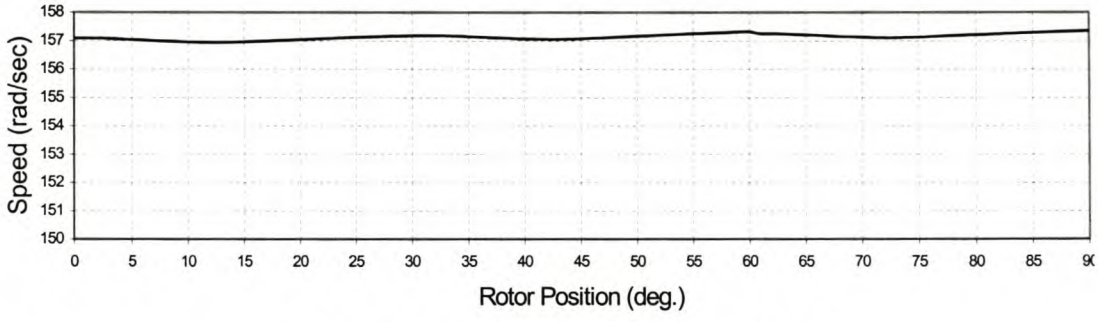


Figure 3.4: Simulated speed variation during SPMO

3.3.2 Current waveform

For the actual SRM operation under motoring SPMO, it is required by [17] that the operation should be with current overlap in order to be efficient and to give maximised motor outputs. Therefore, the adjacent phase currents overlap portrayed in Figure 3.5 indicates the positive outcome in modelling the actual SRM under motoring SPMO by means of the FE analysis. Since we are dealing with a three-phase machine, three phase combinations for the active adjacent phases are available. In mathematical format, two phases conducting current at a time (as it is the case in SRM's actual operation) can be compactly represented as a set consisting of phases x and y, say, [1]. From each of the available SRM phase combinations of the adjacent active phases, this set, can be defined by the relation

$$(x,y)\varepsilon\{(c,a),(a,b),(b,c)\}\ \forall\ T_m \geq 0\ ,\ \omega_m \geq 0\ . \qquad (3.16)$$

The first element in the set denotes the outgoing (switching off) phase and second element denotes the incoming (switching on) phase of the machine as portrayed in Figure 3.5.

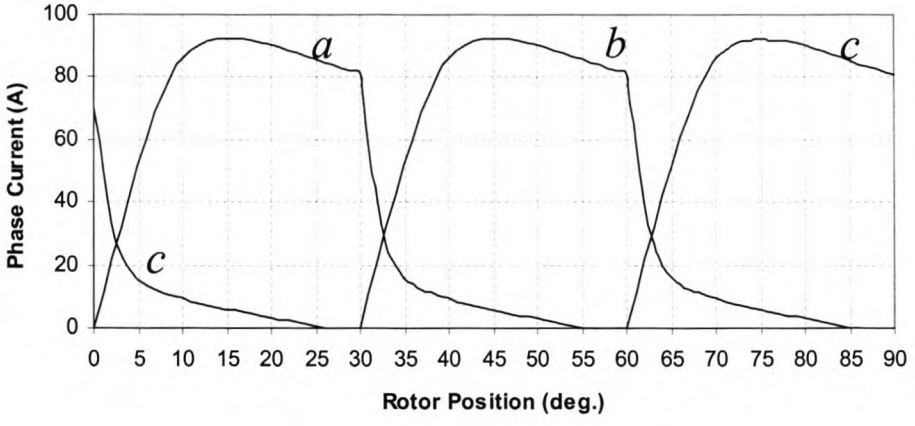


Figure 3.5: Multi-phase current waveform simulations in SPMO

Even though in sections 3.1.1 and 3.1.2 only the machine equations for one of the available combinations of current conducting phases c and a is considered, the current waveform simulation results given in Figure 3.5 incorporates the other remaining two phase combinations, which are (a,b) and (b,c). It should be mentioned that this combination of the phases is true only, say, for first quadrant operation (motoring SPMO) explained in the introductory chapter.

3.3.3 Torque Profile

For every dwell angle, which is 30 *degrees* in this case as shown in Figure 3.5, two adjacent active phase currents contribute to the torque production as portrayed in Figure 3.6. The torque profile portrayed in Figure 3.6 follows directly the current waveform behaviour shown in Figure 3.5, since it drops with the current waveform decay at the current turn-off positions.

Just like in the simulated static torque given in Chapter 2, it could also be seen from Figure 3.6 that there are irregularities in the peak torque. These irregularities are due

to the inadequate meshing of the stator and the rotor poles. In reality the torque profile is smooth.

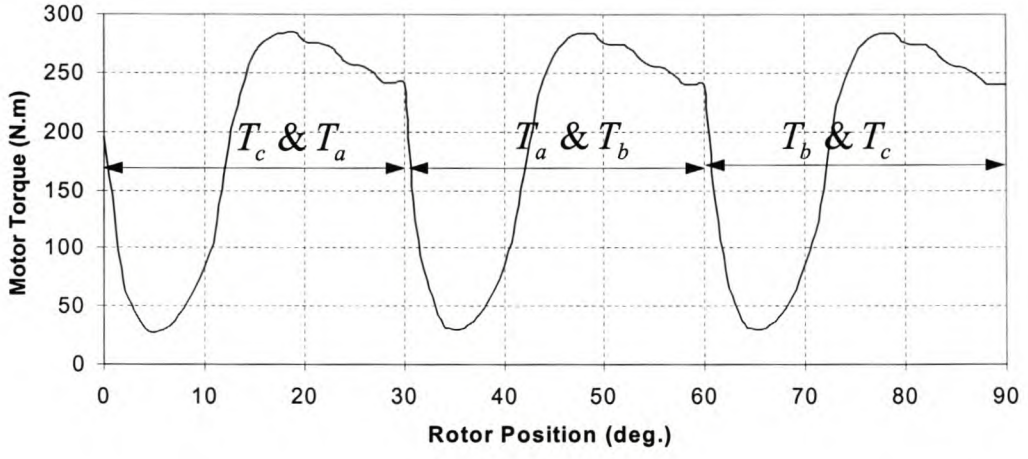


Figure 3.6: Multi-phase Torque versus Rotor Position

3.3.4 Phase current and Self inductance

Figure 3.7 shows the phase a current together with the phase instantaneous-self inductance profile. Phase a is chosen for no particular specific reason either than to illustrate the theoretical results. The results portrayed in Figure 3.7 can be clearly explained with the aid of Figure 3.8, which is the flux-linkage-to-phase-current characteristic-curve. In Figure 3.8 the locus shown by means of the dotted line represents the phase current behaviour from unaligned to aligned positions under SPMO. According to the current behaviour, from one position to another, fairly constant instantaneous inductance, $\frac{\partial \lambda}{\partial i}$, is attained from point A to B as can be observed from Figure 3.8. This accounts for the almost constant inductance behaviour in Figure 3.7, which exists before the rotor and the stator poles overlap. Since the switch-on angle is at the unaligned position, where the active stator phase inductance is low, this is a sufficient phenomenon to justify the phase current drastic increase at the initial positions as shown in Figure 3.7. After the switch-off position (just before the rotor and the stator poles overlap) the phase current drastically decreases and the instantaneous inductance drastically increases as can be explained from point B to A

of Figure 3.8. As seen in Figure 3.7, after the completely aligned position (at 45 *degrees* as shown in Appendix B), the instantaneous inductance again gradually decreases as the rotor passes the aligned position.

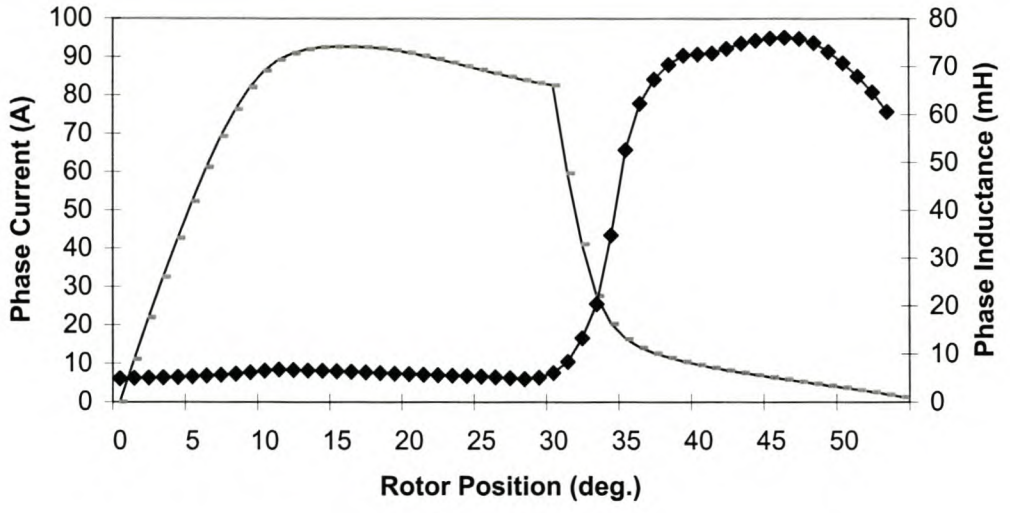


Figure 3.7: Phase current and phase inductance profiles

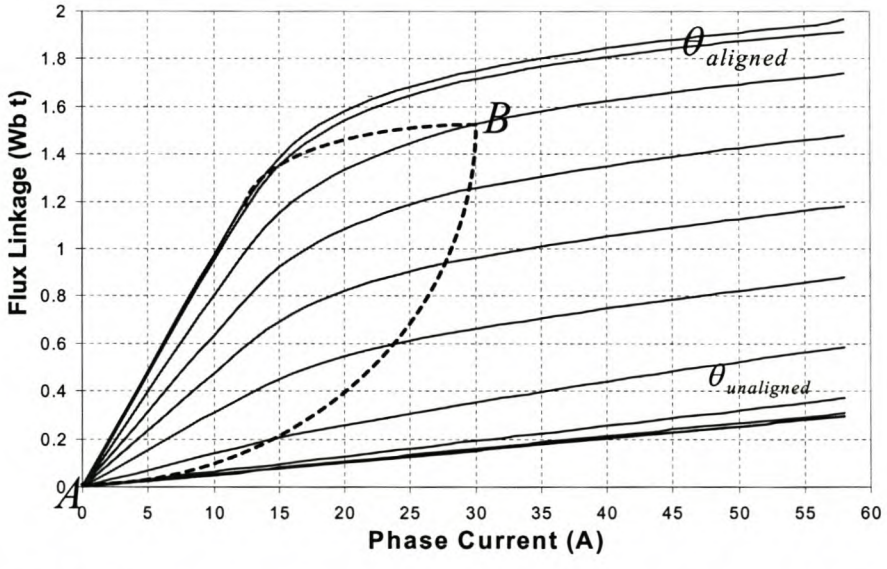


Figure 3.8: Flux-linkage-to-phase-current locus (dotted line) from unaligned to aligned positions

3.3.5 Mutual inductance

In Figure 3.9 a mutual inductance, M_{ca}, for the active phases *c* and *a* is shown. This mutual inductance is observed to be extremely small. For further verification of these

results, in the next section it will be determined how much effect does mutual inductance has on the single pulse mode current waveform simulation results.

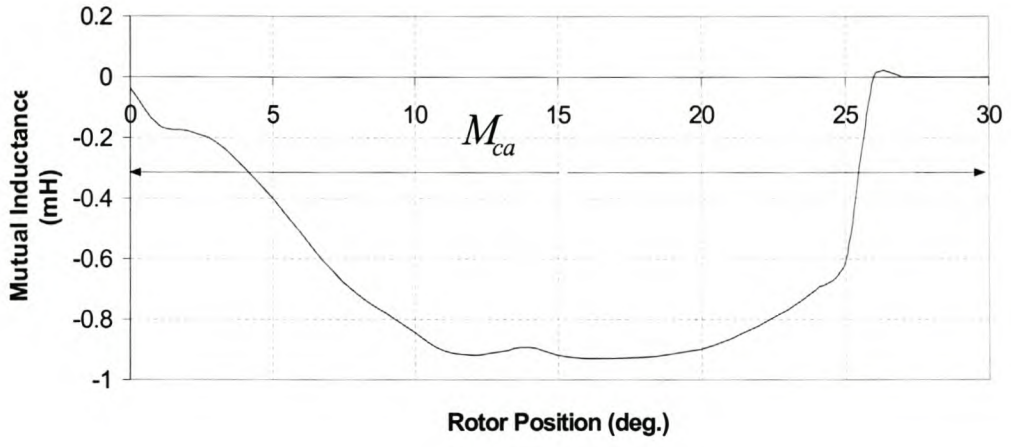


Figure 3.9: Mutual inductances between outgoing and incoming phases

3.4 Other investigations

In this section the simulated current waveform under multi-phase active conditions is compared with the simulated current waveform under single phase active conditions. The aim is to evaluate the level of the effect of mutual coupling. Secondly, the simulated current waveform using the partial differential method is compared with the simulated current waveform using the polynomial method; this is to evaluate the correlation between these simulation results.

3.4.1 Effect of mutual coupling

Figure 3.10 shows the simulated current waveforms under multi-phase active conditions (shown by a solid line) and the simulated current waveform under single-phase-active conditions (shown by the dotted line). From this figure it is apparent that the mutual coupling has almost no effect on the simulated current waveforms. These results fully agree with the mutual inductance profile results portrayed in Figure 3.9. From [1], this is quite a real positive outcome for a well-designed actual SRM.

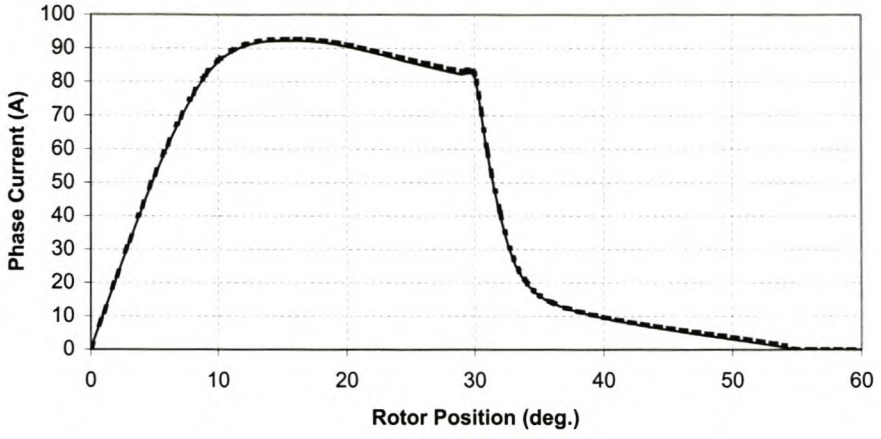


Figure 3.10: Simulated current waveform under multi-phase active conditions (solid line) and single-phase active conditions (dotted line)

3.4.2 Comparison of the simulation-method results

Shown in Figure 3.11 are the single-phase simulated current waveform results from partial differential method of simulation (shown by a doted line) and the polynomial method of simulation (shown by a solid line). A very small discrepancy of approximately 2 A is observed at the peak current waveform position. This implies that the calculated phase current solutions using the two simulation methods have an approximation of 98% in good agreement with each other. Furthermore, the doted current waveform has a more sharp decay compared to the other current waveform. Justification to this discrepancy might be due to the difference in the accuracy of the two distinct mathematical methods used. The small difference between the current waveforms resulting from the two completely different simulation methods is a positive outcome for this investigation.

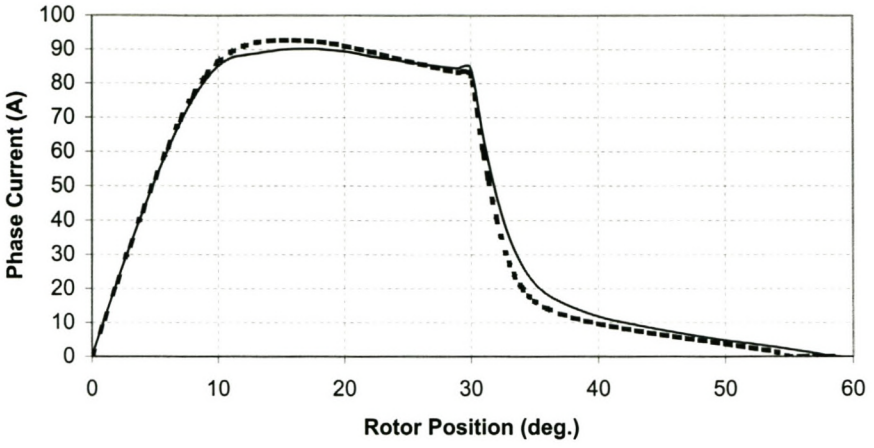


Figure 3.11: Simulated current waveform using the polynomial method (solid line) is compared with the partial differential method (dotted line)

Chapter 4

4 Factors Affecting Simulation Results

In this chapter several factors affecting the simulated current waveform and other simulated machine characteristic results, are given and briefly discussed. Discussions are mainly based on a simple case for one active phase under SPMO. The mesh, the integration accuracy, the switching delay, and the B-H data are the factors whose effect is investigated in this chapter. Since these factors similarly affect the simulation results for both simulation methods discussed in Chapters 2 and 3, any of the simulation methods can be used, which is the polynomial method of simulation in this case.

4.1 The mesh effect

In this section the effect of the mesh on the simulation results is shown. The simulation results using the less dense old mesh are compared with the simulation results using the high denser new mesh.

4.1.1 Less denser mesh

The SRM simulation results given in the previous chapters are based on the mesh structure shown in Figure 4.1. As mentioned in the previous chapters, the irregularities in the peak torque of a torque profile shown in Figure 4.2 are due to the inadequate meshing of the stator and rotor poles. To overcome such irregularities an effort to improve the mesh is made.

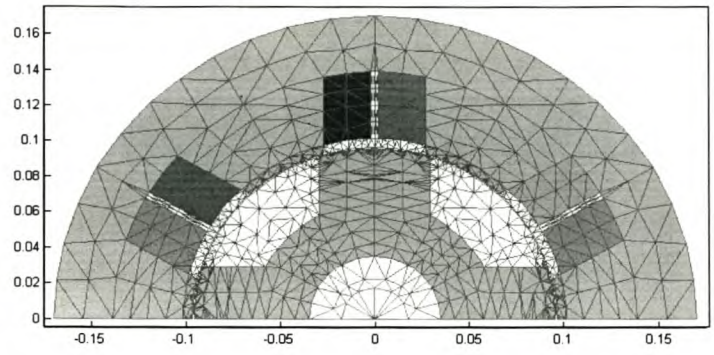


Figure 4.1: Less dense meshed diagram of a tapered stator pole SRM

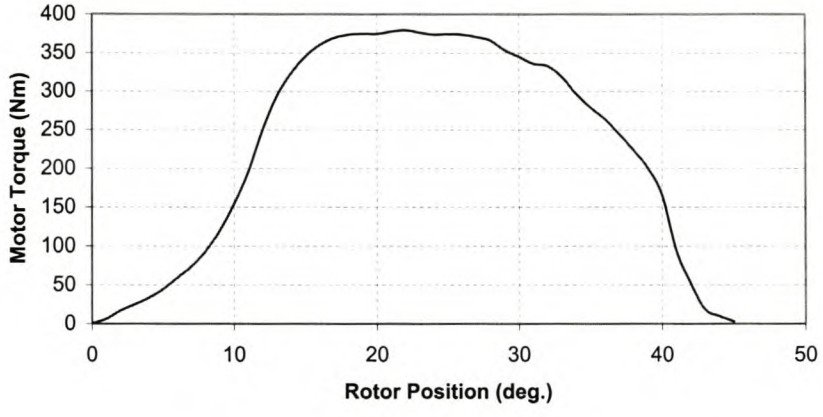


Figure 4.2: Old mesh single-phase static torque profile

4.1.2 High denser mesh

The accuracy of the finite element solution is dependent on the mesh topology. The mesh is thus an important part of the finite element model and lot of attention should be placed on creating it [6]. This is the main reason why it is required to put some effort in improving the mesh topology so as to improve the quality of the simulation results. In the FE package used in this study, the mesh is generated by defining all the nodes in the machine. Depending on the flux distribution, some regions in the motor require more nodes than the others. Only one stator slot of the machine is outlined in terms of xy-coordinates and meshed and mirrored to the number of slots over a pole pitch. The phase windings are then allocated to the slots. Also, one half of the pole of the rotor is outlined and meshed and mirrored to the other half pole (see Appendix B). The stator and the rotor meshes are then joined. The new meshed structure is shown in Figure 4.3

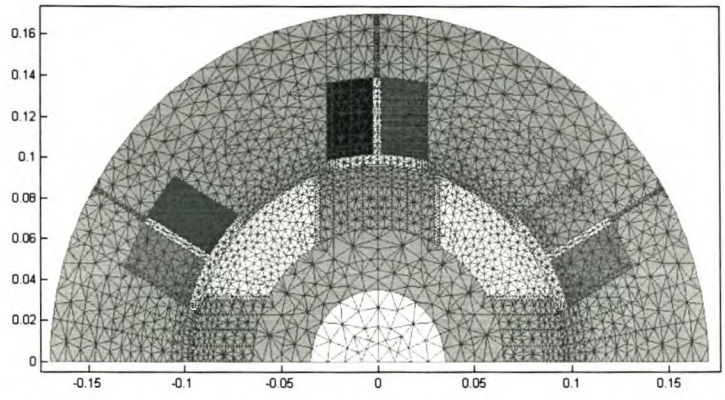


Figure 4.3: More dense meshed diagram of a tapered stator pole SRM

A comparison between the old mesh simulated current waveform (dotted line) and the new mesh simulated current waveform (solid line) is shown in Figure 4.4. Since these results are acquired employing the same simulation method under the same conditions, the justifiable reason to account for the discrepancy at the peaks of the two current waveforms is the mesh topology difference. Furthermore, the resulting smooth torque profile for the new mesh, shown in Figure 4.5 illustrates that the FE solution for the new denser mesh is more accurate compared to the FE solution for the less denser old mesh.

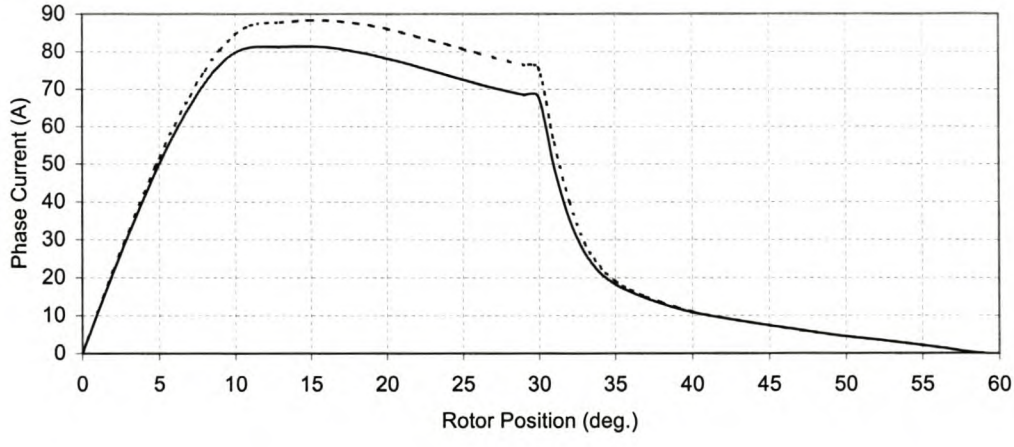


Figure 4.4: Single phase active current waveform resulting from a less dense (dotted line) and a more dense (solid line) meshed structure

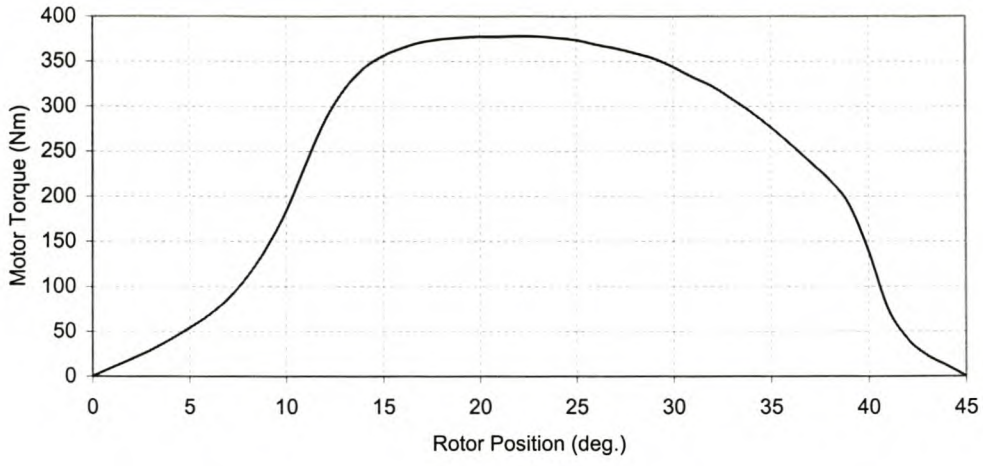


Figure 4.5: New mesh single phase torque profile

4.2 Integration accuracy

As mentioned in the previous chapters, an integration strategy that has been employed to get our simulation results is the Euler integration. Due to this integration method's assumptions that are not true in general (see Appendix A), it is necessary to investigate the extent of the reliability of this integration strategy on the simulation results. Basically, Euler's method for equally spaced inputs produces numerical values of an approximate numerical solution [18]. These values result to a graph of an approximate solution function, say $\lambda = f(t)$ for the differential equation, $v(t) = iR + \lambda'(t)$. For this investigation, verification for the accuracy of the Euler integration is done by checking current waveforms resulting from different Euler's time or position steps (See Figure A1 of Appendix A). It is observed from Figure 4.6 that for the *one*-to-*three degrees* step sizes the current waveforms are observed to be almost on top of each other. The Euler integration has a tendency to lose accuracy as the step size increases. Despite this tendency, it is observed from Figure 4.6 that even *five degrees* position step size doesn't constitute any drastic effect on the resulting current waveform due to the integration inaccuracy. This phenomenon signifies a positive outcome in using Euler integration in the simulation programs.

It is obvious that the bigger the time or position steps used in the simulation program, the shorter the total simulation time becomes. This signifies how reliable it is to use Euler integration in our simulation program as to acquire a variety of options, such as increasing the time or position steps to minimise the total simulation time.

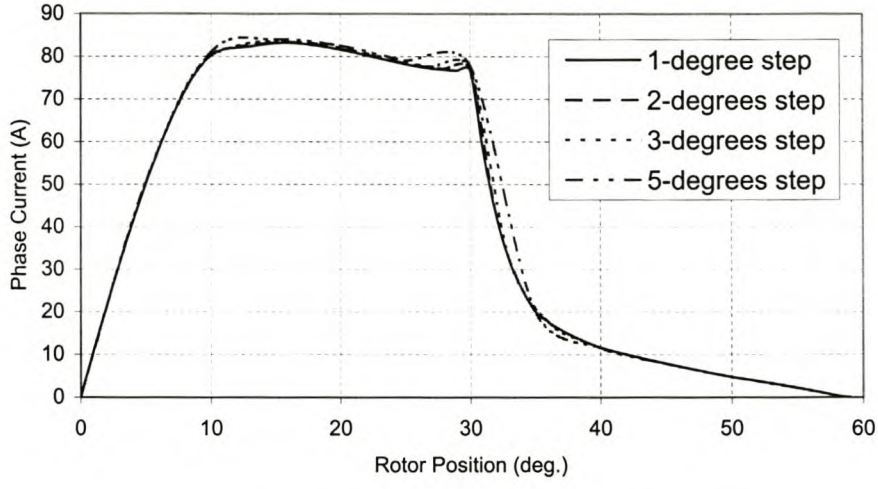


Figure 4.6: Simulated current waveforms using different step sizes

4.3 Switching delay

In a practical scenario it is difficult to accurately select the rotor zero-reference position that is exactly the same as the zero-reference position used in the simulation. This implies that the error in the reference position can easily be *one degree* in a practical situation. Therefore, it is important to investigate the extent in which this error might have an effet on the current waveform under SPMO. Shown in Figure 4.7 is a *one degree* switching delay (i.e. to switch-on the phase winding *one degree* later from the zero-reference position) effect on the simulated current waveform under SPMO. It is observed that even such a small error imposes a drastic effect on the current waveform and hence torque prediction.

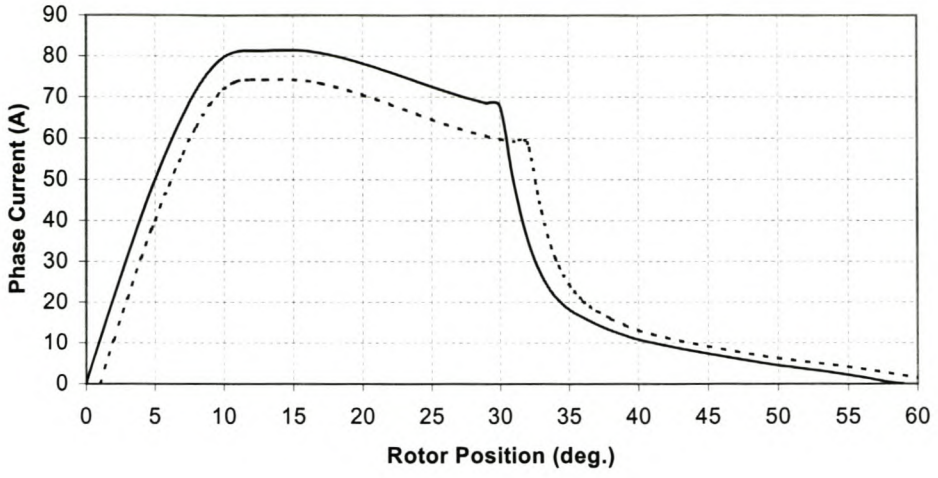


Figure 4.7: 1-degree switching delay effect

4.4 B-H data effect

This section deals with the effect of the B-H data of two different steel materials on the current waveform simulation results. The current waveform behaviour in response to the different B-H data is explained.

4.4.1 B-H curve

Different steel materials exhibit different B-H data. In this investigation, the B-H data of two different steel materials are used as shown in Figure 4.8.

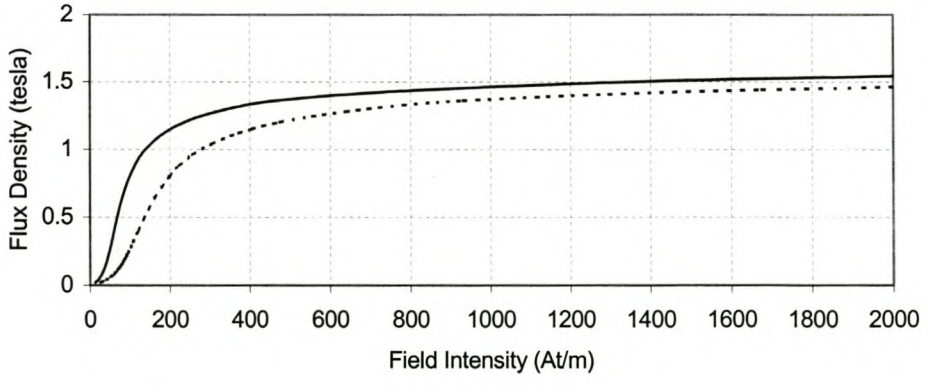


Figure 4.8: Two B-H characteristic curves used in the FE analysis

4.4.2 Effect of the B-H curve on the simulated current waveform

From Figure 4.9 it is observed that during the initial rotor positions (0 to approximately 12 *degrees*), the different B-H data of the different steel materials have very little or no effect on the active phase current. These are unaligned positions with a large air gap, and therefore a high reluctance.The latter hinders the flux from saturating the core. As a consequence of this, there is a small flux variation in the core. This results to a small flux linkage variation, which from the voltage equation

$$v = Ri + L\frac{di}{dt} + \frac{\partial\lambda(\theta,i)}{\partial\theta}\omega \tag{4.1}$$

implies a small systems back emf, $\frac{\partial\lambda(\theta,i)}{\partial\theta}\omega$. Obviously, the inductance term in equation (4.1) is small due to low inductance.

Beyond the 12 *degree* position, when the rotor and the active stator pole start to overlap as shown in Appendix B (for 15-*degrees* position), the effect of the different steel materials with the different B-H data is observed. Truly speaking, it is difficult to explain the cause of this difference directly with the help of the voltage first order differential equation. At this point in the motoring SPMO, the active phase inductance gradually increases as the rotor pole approaches complete alignment with the active phase stator pole. By Ampere's circuit law, there is a direct proportionality relationship between the current, i, and the field intensity, H. Therefore, from Figure 4.8, it is observed that for each particular current value (hence, H value), in the case of the steel material with a higher B-H curve there is a higher flux density, hence a higher flux. This implies that a higher flux linkage is experienced in the active stator phase winding compared to the lower B-H curve steel material. Therefore, the steel material with the higher B-H curve results to the higher flux linkage variation. The higher flux linkage variation in the case of the higher B-H curve implies a higher $\frac{\partial\lambda}{\partial\theta}$, which implies a higher back. A higher back emf will oppose more current in the case of the higher B-H curve steel material. This justifies the resulting lower current waveform (shown by a solid line) in Figure 4.9.

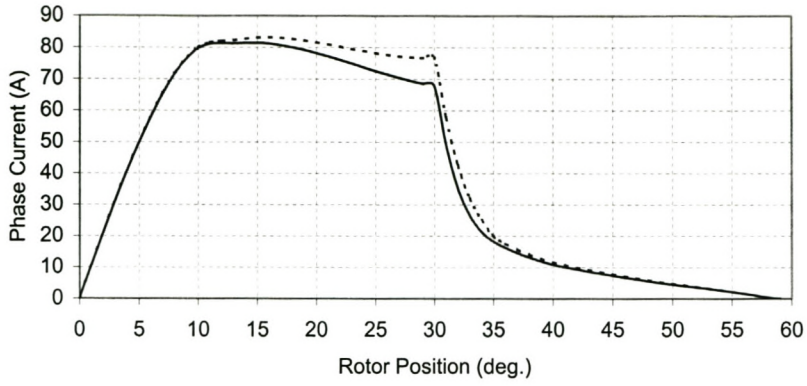


Figure 4.9: Comparison of the simulated current waveforms resulting from the low B-H curve (dotted line) with the current waveform resulting from the higher B-H curve (solid line)

Chapter 5

5 SRM Drive and DSP Control

This chapter shows the hardware set up for the SRM drive. The other machine drive parts are briefly discussed with the exception of the suitable converter topology and the digital signal processor controller that are discussed in more detail.

5.1 Hardware

The hardware set up for the drive is shown in Figure 5.1. The system is divided into four basic parts. These parts are the Supply and Rectifier, Converter, Controller and the Electrical Machine (SRM). The drive is fed from an AC supply which is rectified to give the converter a constant DC bus voltage. The converter with digital controller then generates the three-phase voltages for the machine. As shown in the Figure 5.1, the DSP controller receives the position signal as its feedback signal, since the most important feedback information required by the controller in SPMO is about the position of the rotor. The DSP controller can handle all relevant signals for the efficient control of the system.

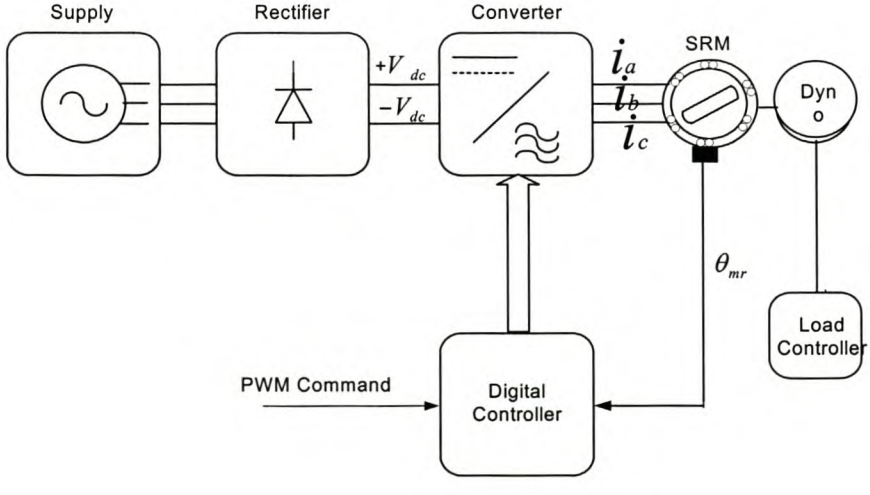


Figure 5.1: Complete SRM drive

5.2 Supply and Rectifier

The supply to the drive system is 400 V line-line AC voltage supply. This supply is able to give up to 400 A of line current. The rectifier used is a three-phase AC to DC passive rectifier shown in Figure 5.2, with the rated power of about 350 kW. This rectifier rectifies the 400 V AC voltage to 540 V DC voltage, which is more than enough for the switched reluctance motor operation. This DC bus voltage is fed through to the capacitor bank and the converter.

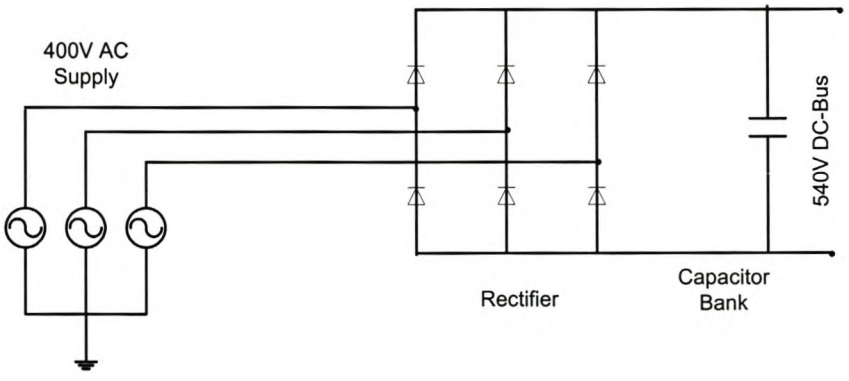


Figure 5.1: Supply and rectifier circuit

5.3 Electric Machine

The electric machine used in this research is a 50 kW tapered stator pole SRM given and discussed in Chapters 1 and 2. Unlike conventional AC and DC machines, the SRM cannot be simply plugged to a DC or an AC source and be expected to run, since the machine has to be switched according to the rotor position.

5.4 Requirements for SRM converter

5.4.1 Basic requirements

There are certain basic requirements that a converter has to meet to supply power to a SR machine. The basic requirements are listed below:

- Each phase of a SRM should be able to conduct independent of the other phases.
- The converter should be able to demagnetise the phase before it steps into the generating region if the machine is operating as a motor and should be able to excite the phase before it steps into the generating region if operated as a generator.

5.4.2 Additional requirements

In order to improve the performance of the machine such as to have higher efficiency, faster demagnetisation, fault-tolerance etc., the converter must fulfil some additional requirements. Some of these requirements are listed below:

- The converter should be able to energise another phase before the off going phase is demagnetised.
- The converter should be able to utilise the demagnetisation energy from the off going phase in a useful way by either feeding it back to the source or using it in the next conducting phase.
- In order to make the commutation period small, the converter must be able to demagnetise the off-going phase in a very short time.
- The converter should be able to freewheel during the chopping period to reduce the switching frequency.

The classic bridge converter, shown in Figure 5.3 was proven to be the most versatile converter topology in modern SRM applications.

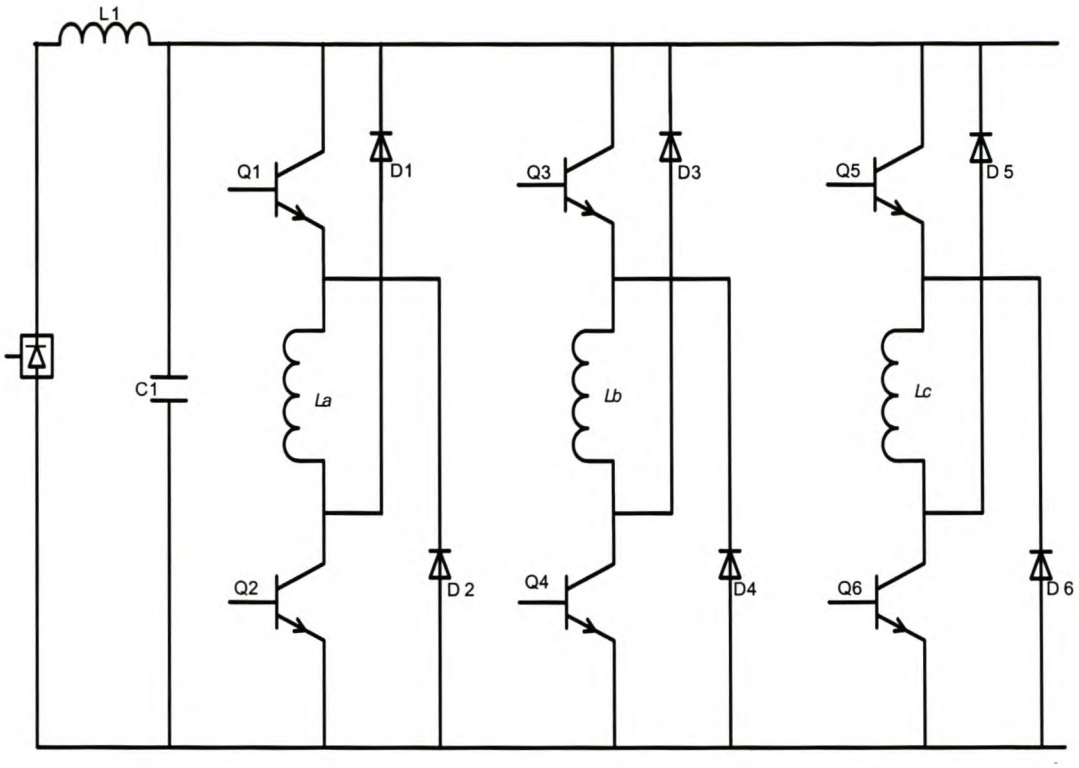


Figure 5.2: Three-phase 2n converter topology using two transistors per phase

Figure 5.3 above shows a schematic diagram of a three-phase 2n topology converter (where n = number of phases) for a three phase SRM. The motor winding is in series with both switches, providing valuable protection against shoot-through faults [6]. Each switch is rated to withstand the DC supply voltage. This converter has several benefits over other SRM converters [14], especially since all the phases are electrically independent. Moreover, the converter part in a SRM drive is capable of four-quadrant operation, with unipolar voltage switching (8.0 kHz effective switching frequency).

5.4.3 Classic converter operation

There are four modes of operation in a classic converter chosen for this research. During the conduction mode of phase *a* (*La*) shown in Figure 5.3, both phase leg switches (Q1 and Q2) are in an on state. The input DC source magnetises the phase. This mode is usually initiated before the start of the rotor and the stator pole overlap, so that the phase current reaches the reference value before the phase inductance

begins to increase. This helps to reduce the torque ripple. When the current reaches the reference value the converter steps into the regulation mode. In this mode the current is controlled to a reference value by switching one of the phase switches while leaving the other one on till the commutation time is reached. This type of regulation strategy is referred to as 'soft chopping'. Only $+V_{dc}$ and zero are used in soft chopping, which often suffices for motor operation as the required voltage is always positive for turn on and turn off of a phase [15].

Both of the phase switches are turned off to initiate commutation. The phase starts to demagnetise through the two freewheel diodes and the energy transfers from the motor phase to the dc source. During commutation the off going phase winding sees a voltage of $-V_{dc}$. While one phase is demagnetising, another phase can be magnetised. This helps reduce the torque ripple during the commutation. The main advantage of the circuit in Figure 5.3 is the rate of demagnetisation when both switches are inactive. This state will cause the complete bus voltage to appear reversible over the phase winding. The control of this system is simplified by an excess energy that is forced into C1 from where it is forced back into a different phase without any control system to redirect the flow of this energy. This is the best factor that makes the classic converter a popular choice in the modern SRM applications.

5.5 Digital Controller

Between the two DSP categories given in [10], the fixed-point processor is the one used in this project. Within this category, there are DSP's for specific use, like for motor drive controllers. These DSP's have built in analog-to-digital converters, PWM outputs and much more for specific use. The DSP that is chosen for the SRM drive system is part of the TMS320C24* Series. The TMS320F240 DSP, which is used has few features, namely:

- 16K Flash RAM
- 12 Compare/PWM channels
- Three 16-Bit General-Purpose Timers
- Dual 10-Bit Analog-to-Digital conversion module

- Serial Communications interface module
- Serial Peripheral Interface Module
- 6 External Interrupts (Inc. Power Drive Protect Interrupt)
- 50 ns Instruction Cycle Time

In SPMO there is no need for current and voltage feedback. Therefore, the current and the voltage feedbacks into the DSP processor are thus ignored. As a result, out of four blocks of the control unit, three blocks are used in this drive as shown in Figure 5.4. The first and the main block is the processor/protection block, which hosts the DSP processor. The other two blocks are the fibre optic interface connected to the converter driver circuits by means of the fibre optic cables and the rotor position measurement interface. The fourth block left out in this drive mode of operation (SPMO) is the current and the voltage measurement card. Each of the blocks is built on its own circuit board. All the circuit boards are properly fitted into the euro rack configuration cabinet as shown in Figure 5.5.

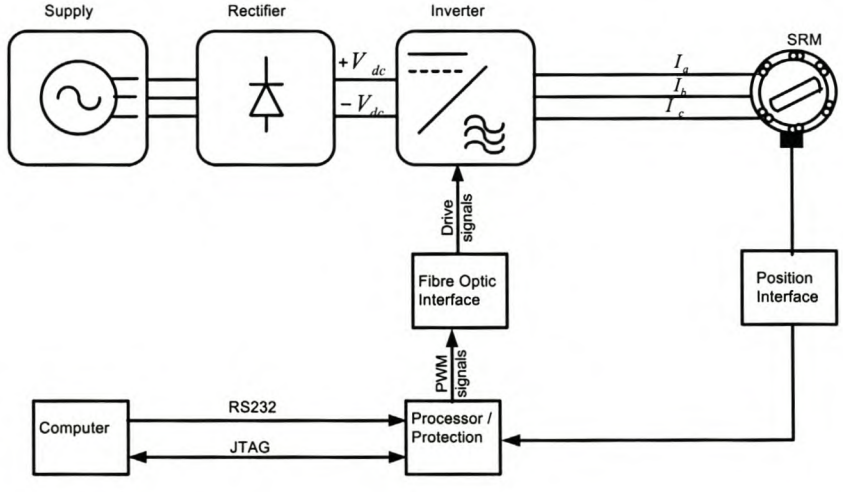


Figure 5.3: DSP Control Unit

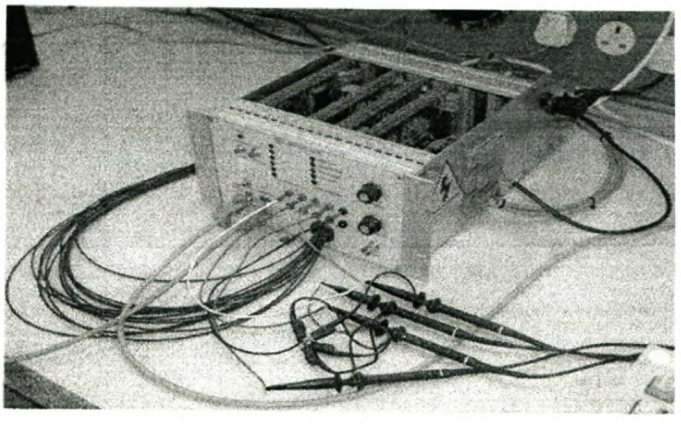


Figure 5.4: DSP Controller

5.4.4 Processor/Protection Card

As mentioned in the previous section, the processor is the main card of the controller. The DSP is connected through a data bus with the other cards. This card contains the following parts:

- A 20 MHz Fixed Point Digital Signal Processor
- A Programmable Logic device (EPLD)
- Four Digital-to-Analog Outputs
- RS232 Communication Port
- JTAG Communication Port

A layout of the processor/protection card is given and summarised by the flowchart diagram given in figure 5.6

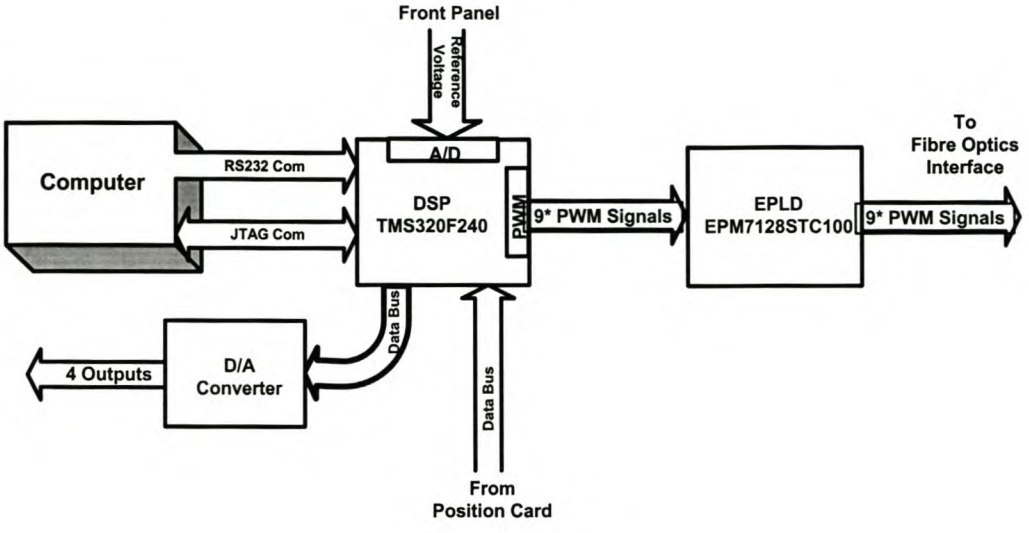


Figure 5.5: Layout of Processor/Protection Card

a. Communication interface (JTAG and RS232)

The processor has two communication ports through which it can be programmed. These are JTAG communication port and the serial port (RS232 port). These serve as a communication link between the computer control software program and the DSP. Due to its direct access to the FLASH RAM and the DSP registers, the JTAG port is used to initialise the processor and to load its serial port's loader program in the FLASH RAM of the DSP. The second communication port, called the serial port is used to download the control software into the DSP.

Again, the user should be able to access some information given by the processor about what is happening and at what point in the program the processor is. For this, there are two types of inputs and outputs available, namely:

- standard I/O pins to the processor that is accessible from the front panel of the controller.
- four analog channels from DAC's connected to the DSP data bus.

With the help of these analog channels every variable or counter inside the DSP can be monitored for debugging.

b. The Processor Circuit (DSP TMS320F240)

As shown in figure 5.6 the processor circuit mainly consists of the DSP processor together with its inputs and outputs as well as the communication ports. Most of the inputs to the DSP are digital with three I/O ports available. At the back plane of the euro rack the 16-bit data bus and address bus are available for use by the position card. The DSP used in this project has sixteen analog-to-digital channels that are connected via the universal bus to the other interface cards of the controller. All the measurements as well as the inputs from the user are connected to these channels. The channels are divided into two groups. The first eight channels are connected to the first ADC and the remaining eight are connected to the second ADC. These 16 channels are multiplexed through two ADCs. The control of the ADCs is done with the DSP software program. Since the soft switching strategy of the classic converter is employed in this project and there is no dead time required according to the converter topology, three from six PWM pins of the full compare and another three pins from standalone pins explained in [10] have been used. PWM signals are connected through the protection algorithm of the EPLD (see Figure 5.6) and in case of fault the PWM signals can also be disabled there.

c. Protection Circuit (EPLD EPM7128 STC 100)

This protection circuit consists of the programmable logic device (EPLD) as shown in Figure 5.6 with six PWM signals. By using a selective six signals, either six (Bipolar switching) or twelve (Unipolar switching) signals can be given to the converter. This logic controller receives error from the Fibre Optic and the position interfaces. One of the errors is from the converter. On the above error, the EPLD will immediately switch off all the drive signals to the converter.

5.4.5 Position Card

This serves as a second measurement card of the control unit. An externally mounted resolver on the rotor of the SRM generates the rotor position feedback signal. This resolver supplies the resolver card of the DSP with encoded position information. The resolver controller circuit can be selected to give different resolutions of position. It can give 10-, 12-, 14-, or 16-bit resolution. For this application the 12-bit digital position is directly read into the data bus of the DSP. The position value is read from the resolver program position reading part in the C-code in Appendix C. This is then saved as rotor angle. The 12-bit accurate resolver is converted to a 16-bit value for the DSP calculations. A fixed angle is added to the rotor angle to line up the zero position of the rotor to the correct zero position with the resolver card. The final part of the program shows the switching strategy for a conduction mechanical angle of 30 degrees from zero to 360 degrees.

5.4.6 Fibre Optic Interface Card

This acts as an interface between the controller and the converter. This interface isolates the controller and the converter electrically. The controller signals that switch different IGBTs are transmitted via optic light through the optic cables from the controller to the converter. The card converts twelve or six drive signals from digital electric signals to optic signals. On the converter side the inverse is done to the switching states for the IGBTs. There are also three optic error signals coming from the converter. They are received by the interface card and combined into a single electric error signal that is sent to the protection card.

Chapter 6

6 Comparison of Measured and Simulated Results

Static torque measured results were done in the previous SRM research done in [6]. In this chapter, these results are given and compared to the more accurate static torque calculated results from the new mesh given in Chapter 4, to verify the degree of correlation in terms of the shape of the torque profiles as well as their peak levels. The running tests using the tapered stator pole prototype SRM available from the previous research is also dealt with, to attain multi-phase current waveform results under motoring SPMO. A comparison of the FEM multi-phase current waveform calculated results with the measured multi-phase current waveform results under motoring SPMO is made to determine the extent of agreement between the two results. Finally, the simulation time for the multi-phase simulated current waveforms is checked for both the old and the new meshed structures using an 800 MHz and a 1.67 GHz computer. This is aimed at highlighting and exposing the reasonability of this simulation package in terms of the simulation time with the computer speed improvement from old to new computers.

6.1 SRM drive and test system

This section briefly reviews the SRM drive system explained in chapter 5. The static torque measured results from the previous investigation done in [6] are compared with simulated static torque results to verify the level of the accuracy of the FE solution in comparison with the actual systems results. The initial steps taken to set up the drive system in preparation for the running tests are also given. This involves the choice of the machine's zero switching position, which is the most critical and significant step dealt with in order to acquire good results under motoring SPMO.

6.1.1 SRM Drive Overview

The hardware of the SRM drive set up is given in Figure 5.1. A large capacitor bank is added into the system to ensure minimised voltage ripple effect on the main DC link. The machine used for the test results is a tapered stator pole SR machine prototype shown in Figure 6.1.

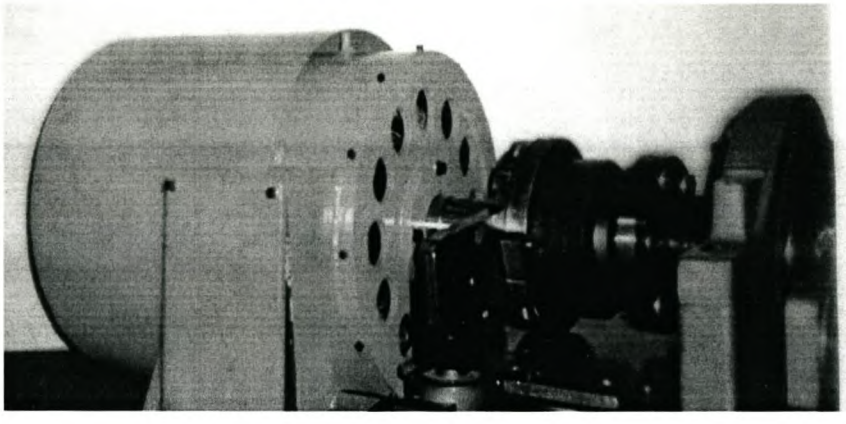


Figure 6.1: Tapered stator pole SRM prototype

6.1.2 Static tests

At this point, we need to recall that the simulation program uses FE solution actively during the simulation process to attain the flux linkages and the motor torque. Therefore, it is a valuable step to confirm the accuracy of the FE solution. Comparing the static torque measured results from the previous research with the calculated results to verify correlation between the two results does such a confirmation. The test set up used for measuring static torque results is shown in Figure 6.2.

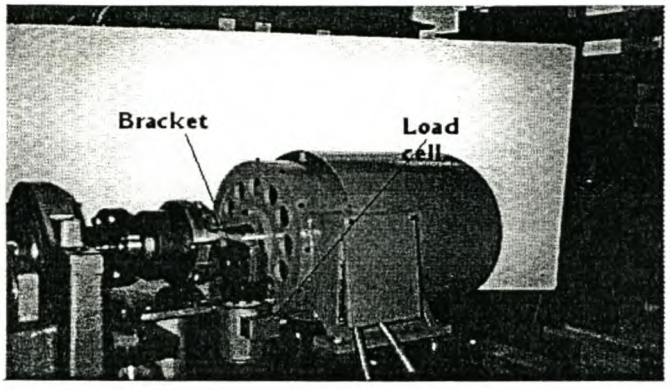


Figure 6.2: Static test set up

The measured and the calculated results are given in Figures 6.3 and 6.4, respectively. From Figure 6.3 it is observed that the phases exhibit similarity in terms of the torque profiles. The torque profile is smooth in reality. The small differences can be attributed to, amongst other things, the imperfection and the inaccuracy of the available measuring equipment. The deeps and the pulsations observed in the measured static torque profiles are also due to some measurement errors. In spite of these uncertain features of the measured results, the calculated and the measured static torque profiles show to some extent an agreement in terms of the shape and peak levels, which signify a positive indication about the accuracy of the FE solution. Again, the acquired smooth simulated static torque profile is a positive indication about the accuracy of the FE solution.

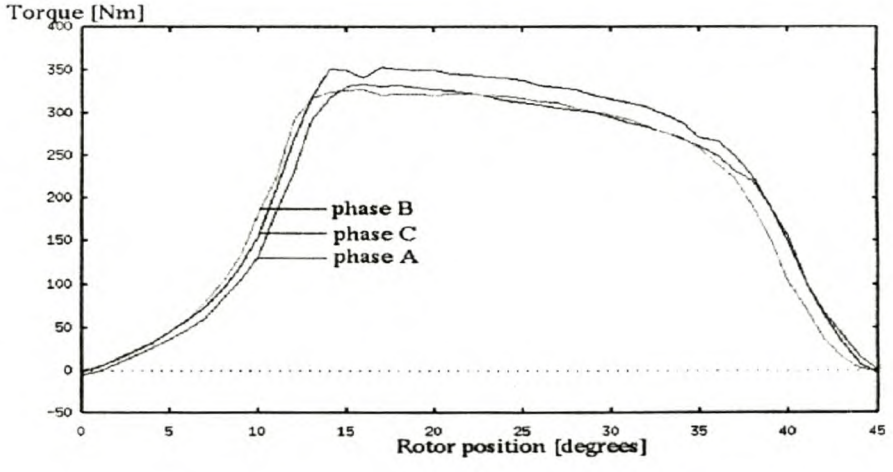


Figure 6.3: SRM measured three-phase static torques

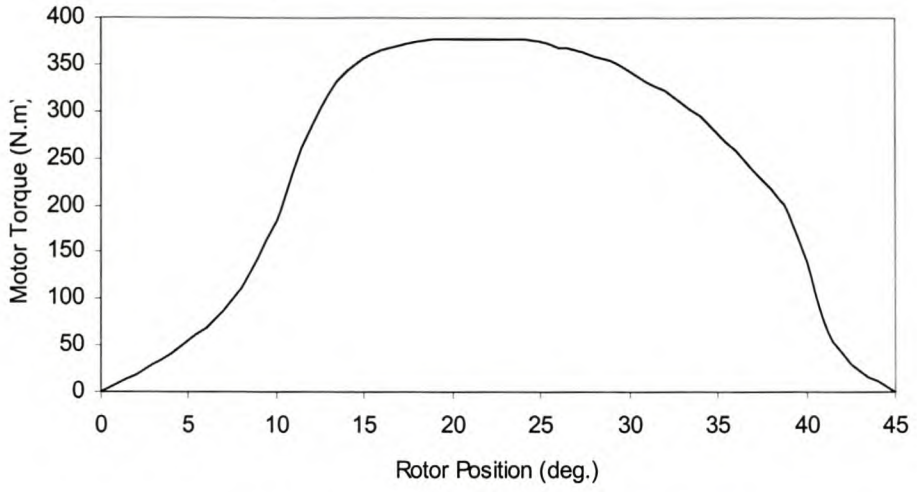


Figure 6.4: SRM simulated one-phase static torques

6.1.3 Running tests set up

As a preliminary step, it is important for the user to check whether there is a proper communication between the computer C-code software program and the DSP. To do this, the reference current and position signals from the C-code program are down loaded to the digital signal processor via the serial communication port. These are then read as analog signals at the front panel of the controller by means of the oscilloscope. Another valuable initial step is to check whether the digital switching signals function according to the proper switching sequence. Observing the digital switching signals on the IGBT terminals at the converter side, by manually rotating the shaft of the machine, does this. The direction of the rotation of the motor depends on the sequence of switching the stator phase windings. The most difficult step in the preparations to get the SRM drive on line is the choice of the starting rotor position. The choice should be made such that it is reasonable with the switching sequence programmed in the C-code given in appendix C. For future results comparison purposes the machine zero position is chosen with reference to the zero position of the FEM field plot diagram given in Appendix B. In response to the chosen zero machine position, in the DSP, that particular position value is read from the resolver as a non-zero value. It should be mentioned without any loss of trend that the mechanical angle for 0 to 360 degrees is equal to 0 to 4095 in the DSP. The offset value read from the resolver is then adjusted to zero by subtracting it from 4095 in the C-code program.

The result is then advanced to the mechanical rotor angle (rotorpos) as shown in the C-code program given in the next page.

```
                    /* Read the position of the rotor */

 *PCDATDIR & = 0xFFFD;

*PCDATDIR & = 0xFFFE;  /* Make IOPC0 (ENABLE) = 0 */
rotorpos = *Datain;
*PCDATDIR = 0x0303;      /* Make IOPC1 (FREEZE) = 1 */
rotorpos  =  rotorpos >> 4;  /* Make position a 16 bit value */

rotorpos &  =  0x0FFF;
rotorpos = rotorpos + 3590;  /* Zero of angle */
if (rotorpos > 4095)
{rotorpos = rotorpos - 4095;}

                        /* Write to the DACs*/
 * PBDATDIR & = 0xFFBF;        /* Select DACs 4*/
 * Datauit          = rotorpos;        /* Write mechanical rotor angle to DAC*/
```

The machine phase current cables are then connected through three 200 A LEM modules used as current sensors. These LEM modules give a current signal with a ratio of 1:2000 for the current sensed by the sensor. Since this is an open loop system, on the basis of the current and the voltage feedback the sensor cables can be supplied between $\pm 15\,V$ using a voltage supply away from the DSP. The signal from the current sensor is then terminated into a 100 Ω and the voltage over the resistor is shown on the oscilloscope. This voltage represents the current waveform of the measured phase.

6.1.4 Running Test Results

Initially, the tests are done at 500 *V* DC bus voltage and 1500 *rpm* speed conditions. The main problem encountered with the preliminary test results is that they are unstable as shown in Figure 6.5 (d). The instability could be attributed to, among other factors, mechanical oscillations and the resonance in the mechanical system. Despite the instability problem, a positive outcome is that, the multi-phase current waveforms shown in Figure 6.5 (d) do overlap as required.

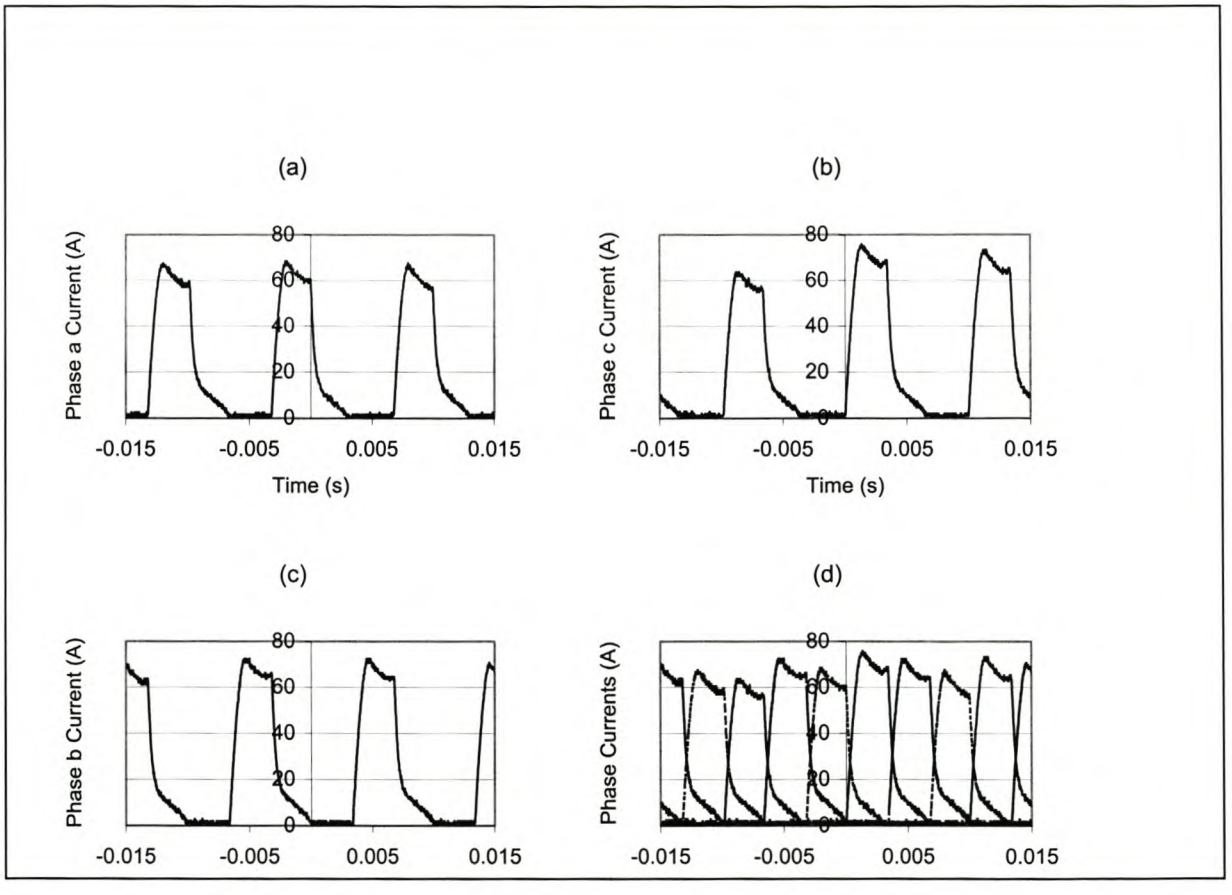


Figure 6.5: SRM phase current waveforms

At this juncture, it is important to determine the other factors that might contribute towards this instability. As shown in Figure 6.6, a 500 *V* DC bus voltage still has some ripple component, despite the high capacitor bank incorporated on the main DC link. This observation enthused the idea of checking the speed signal behaviour under

these conditions. To achieve this, the speed programming is done in order to read the speed signal from the processor front panel. The C-code program for the speed signal is read as follows:

```
/* calculate the speed */

rotorposa = rotorpos;
if (rotorpos < prerotorpos)
{rotorposa = rotorpos+4095;}
speedref = 100*(rotorposa-prerotorpos);
prerotorpos = rotorpos;


/* Write to the DACs */

*PBDATDIR &= 0xFFF7;   /* Select DAC 1 */
*Datauit = Ref;        /* Write Reference to DAC */
*PBDATDIR |= 0x0008;
```

From Figure 6.7 the ripple component is extremely small compared to its corresponding DC bus voltage ripple component shown in Figure 6.6. It is clear, therefore, that unstable DC bus voltage has a major contribution in the instability of the resulting current waveforms than the speed.

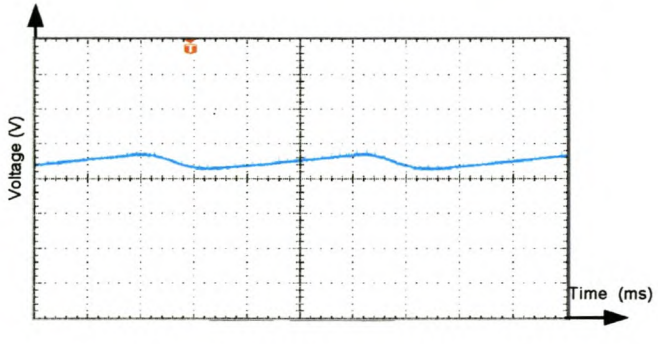


Figure 6.6: DC-link bus voltage variation (at *500 V*)

x-axis: 25 ms/div.; y-axis: 100 V/div.

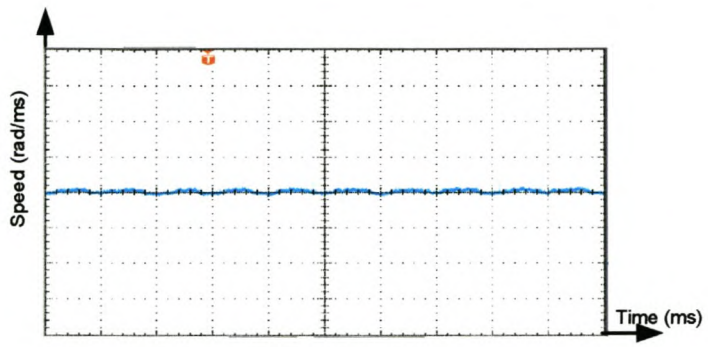


Figure 6.7: Speed under 500 V DC-link voltage

x-axis: 25 ms/div.; y-axis: 375 rpm/div.

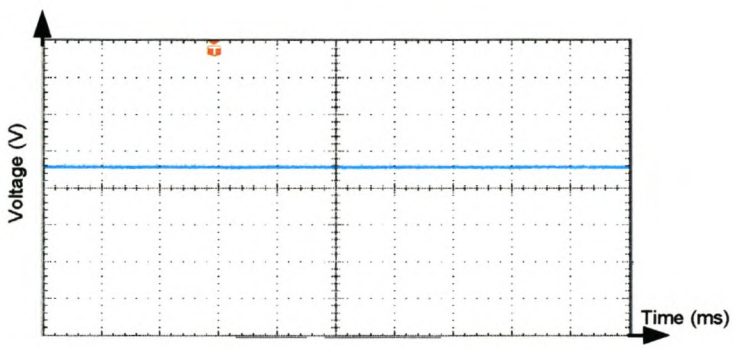


Figure 6.8: DC-link bus voltage variation (at *524 V*)

x-axis: 25 ms/div.; y-axis: 100 V/div

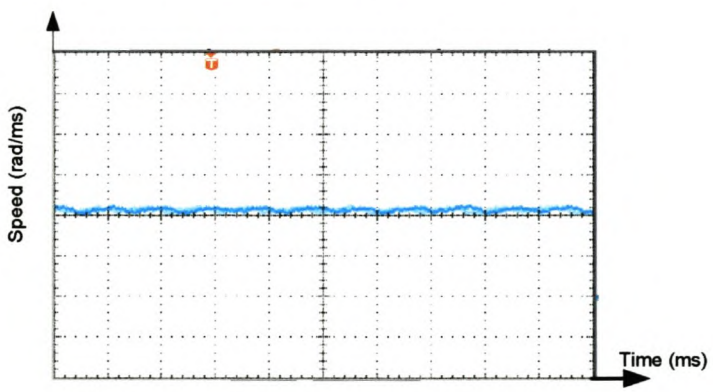


Figure 6.9: Speed under 524 V DC-link voltage

x-axis: 25 ms/div.; y-axis: 375 rpm/div.

Raising the DC bus voltage to 524 V at a speed of 1572 rpm as the new test conditions shows some improvement on the DC bus voltage and speed signals in

terms of the instability as portrayed in Figure 6.8 and Figure 6.9, respectively. Under these conditions, the current waveforms are initially much stable but suddenly lose their stability as shown in Figure 6.10 (d). Since the DC link voltage and the rotor speed are more stable as shown in the Figures 6.8 and 6.9, it follows that such a sudden change in the stability of the phase current waveforms could also be associated with, among other factors, mechanical oscillations and the resonance in the mechanical system. From Figure 6.10 (d) it is observed that the instability is extremely low compared to the instability experienced under the previous conditions as shown in Figure 6.5 (d).

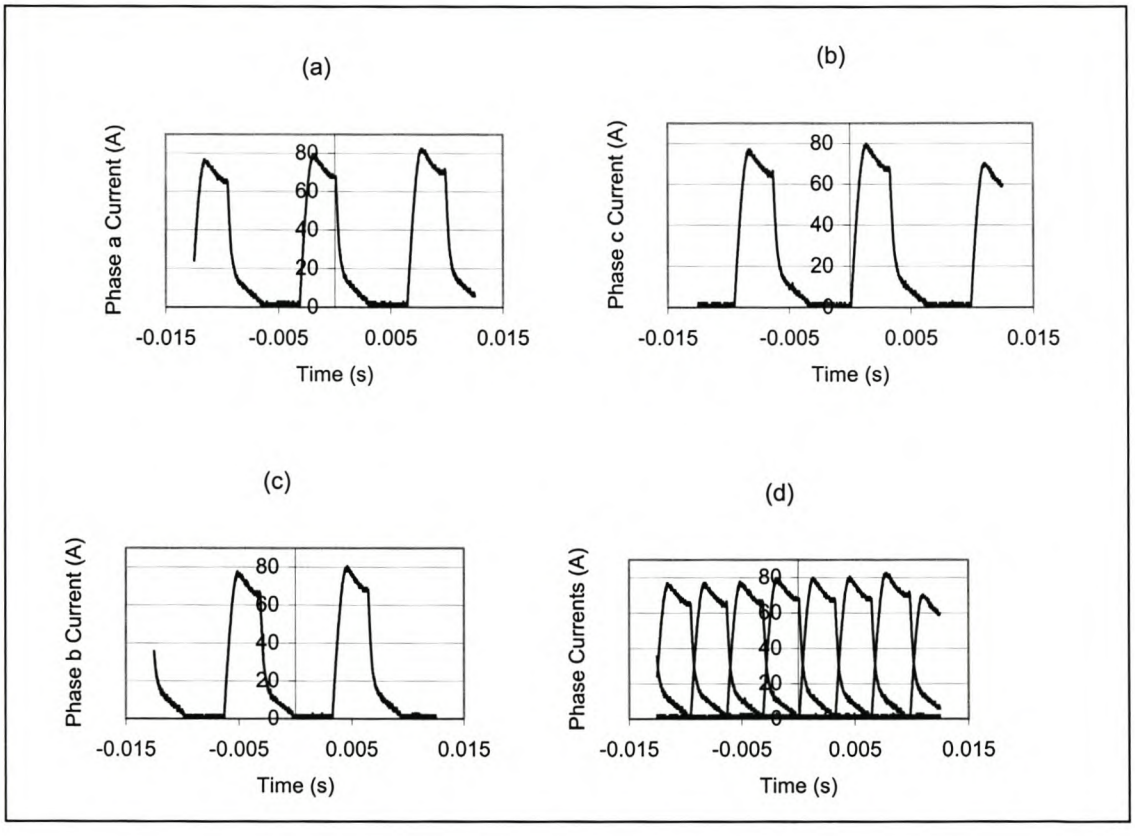


Figure 6.10: SRM current waveforms

6.2 Comparison of the FEM and Measured Results

In this section the comparison between the simulated and the measured multi-phase current waveforms is made to evaluate the degree of correlation between the two results. For both results, a 524 V DC bus voltage at a rotor speed of 1572 rpm is used. The systems moment of inertia equal to 0.8 kg.m^2, is also used in the simulation results. The SRM used is a 50 KW machine described in Chapter 2. The simulation results used for the comparison are based on the new mesh structure shown in Chapter 4.

From Chapter 4, apart from the other factors affecting the SRM phase current, the dwell angle is noticed to have quite drastic effect on the SRM current waveform under motoring SPMO. To give the clear clarification portraying how this factor affects the comparison between the measured and the simulated multi-phase current waveforms, such a comparison is made on the basis of the three switching angle states for the simulated results, namely:

- Measured results compared with no dwell angle delayed simulation results
- Measured results compared with one degree dwell angle delayed simulation results
- Measured results compared with two degrees dwell angle delayed simulation results.

6.2.1 Comparison with no dwell-angle-delay simulation results

The single pulse mode calculated and measured current waveform results are shown in Figure 6.11. The conditions for the measured and simulation results are almost the same, except that the load torque fixed in the simulation results is surely a non-constant value in the case of the practical scenario. The friction coefficient, B, which has been assumed to be zero in the simulation results is a non-zero factor in the practical scenario. In both cases, the phase current overlaps are taken into consideration. This makes the comparison almost valid. The two current waveforms given in Figure 6.11 show to some extent, an agreement. The other possible reasons that could be accountable to the discrepancy between the two current waveforms shown in Figure 6.11 are as follows:

- A 2D FE analysis used in the simulation ignores the actual 3D effects
- Possible dwell angle delay error could easily be committed in a practical situation as will be shown in the next sections.

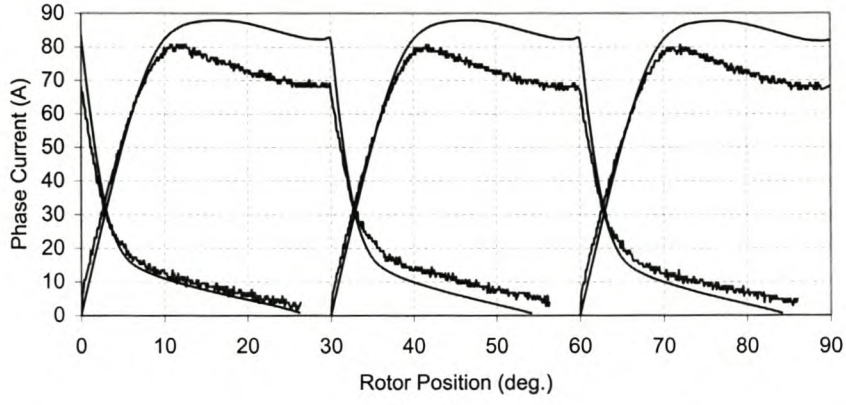


Figure 6.11: Finite element calculated (smooth line) and measured (rough line) current waveforms (zero switching on position for the simulated results)

6.2.2 Comparison with 1-degree dwell-angle-delay simulation results

The results portrayed in Figure 6.12 show the SRM measured current waveform compared with the simulated current waveform results. The simulation conditions are exactly the same as the ones in the previous case, except that 1-degree dwell angle delay is considered in the simulation program.

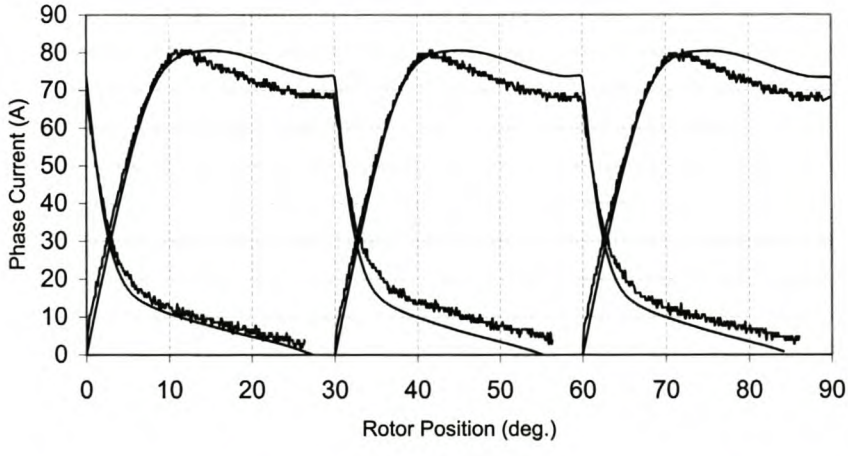


Figure 6.12: Finite element calculated (smooth line) and measured (rough line) current waveforms (1-degree switching on position for the simulated results)

From the figure, it is seen that one degree delayed simulation results are more close to the measured results. Such a small error can easily be made in practice. A possible reason to the small discrepancy in these results could be due to high inductance experienced in the 3D measured compared to the 2D simulated results. The other possible reason that might have contributed towards this discrepancy could be due to the core losses that are ignored in the simulations.

6.2.3 Comparison with 2-degrees dwell-angle-delay simulation results

Final comparison is made between the measured results and 2-degrees dwell angle delayed simulation results as shown in Figure 6.13. The observation that the simulated results are lower than the measured results in Figure 6.13 indicates that the higher the delay error the lower would be the resulting current waveform profile attained.

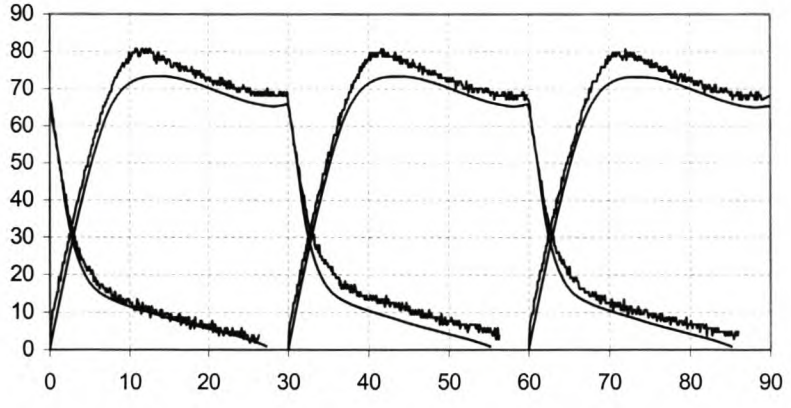


Figure 6.13: Finite element calculated (smooth line) and measured (rough line) current waveforms (2-degrees switching on position for the simulated results)

6.3 Simulation time evaluation

In this section, an attempt to show how the simulation program using the partial differential method of simulation improves in terms of the simulation time with old low speed and new high speed computers is made. Using this simulation method, the simulation time is evaluated employing the old mesh and the more accurate new mesh. 800 MHz and 1.67 GHz computers are used to perform this evaluation. Tables

6.1 and 6.2 show the number of the field solutions, time per field solution and the total simulation time for the less accurate old mesh given in Chapter 4.

Computer speed (*MHz*)	Time step (μs)	Time per field solution (*s*)	Number of field solutions	Total simulation time (*min*)
800	110	50	90	75

Table 6.1: Information about simulation time based on the old mesh

Computer speed (*GHz*)	Time step (μs)	Time per field solution (*s*)	Number of field solutions	Total simulation time (*min.*)
1.67	110	22	90	33

Table 6.2: Information about the simulation time based on the old mesh

Similarly, in Tables 6.3 and 6.4, the same evaluation is also made from the more accurate new mesh using the same 800 *MHz* and 1.67 *GHz* computers.

Computer speed	Time step (μs)	Time per field solution (*s*)	Number of field solutions	Total simulation time (*min*)
800 MHz	110	67	90	100

Table 6.3: Information about the simulation time based on the new mesh

Computer speed	Time step (μs)	Time per field solution (s)	Number of field solutions	Total simulation time (*min*)
1.67	110	30	90	45

Table 6.4: Information about the simulation time based on the new mesh

From these outcome results it is observed that for both less accurate old mesh and the more accurate new mesh, an improvement from 800 *MHz* computers to 1.67 *GHz* computer has reduced the simulation time by a reduction factor of more than *2*. This implies that the simulation speed with the new computer (1.67 *GHz*) is more than double the speed of the old computer (800 *MHz)*. This clearly shows that in 5-10 *years* to come, the same simulation program will take even less than a minute to complete running.

Chapter 7

7 Conclusions and Recommendations

The conclusions drawn are based on the use of the FE analysis to develop two distinct mathematical models to simulate the SRM current waveforms under SPMO. The feasibility of the developed simulation program in terms of the simulation time is also investigated. Recommendations are given to further investigate the FE program simulation results to acquire even better comparison with the SPMO measured results.

7.1 Conclusions

I. The phenomenon that the proposed simulation methods show almost the same results confirm the correctness of the simulated results.

II. The good agreement between the simulated and the measured results further confirm the correctness of the proposed simulation methods.

III. From the simulation time investigation, the following conclusions are drawn:

- The use of the FE solution actively during the simulation is feasible in terms of the simulation time. The reduction of the simulation time by a factor of more than *2* from 800 *MHz* to 1.67 *GHz* computers, confirms the feasibility of the proposed simulation methods. This implies that with the generation of the fast computers the simulation time will be drastically reduced. Hence the proposed simulation methods in this thesis will become more and more feasible.
- The density of the mesh plays an important role in the simulation time (see recommendations).
- The method of numerical integration used and the simulation time or position step-length also plays an important role in the simulation time (see recommendations).

IV. The density of the mesh used in the FE analysis has a huge effect on the accuracy of the FE solution that is directly used by the simulation program. Care has to be taken to maintain this accuracy. One way to check the accuracy of the FE is to look at the static torque profile if it is smooth.

V. It has been shown that the current waveform under SPMO is very sensitive to the switch-on angle. Hence, the zero reference position must be determined very accurately in the practical SRM drive system.

VI. Comparison of the single phase current waveform under multi-phase excited simulation condition with the single phase current waveform under single phase excited condition are the same. This behaviour shows that mutual coupling has negligible small effect on the current waveform results for this particular machine design. This behaviour, however, might differ from one design to another. In the case of the design with no mutual coupling effect it suffices to use only one excited phase in the simulations.

VII. The B-H data used in the nonlinear FE analysis shows to have an effect on the current waveform. It is therefore strongly recommended that the exact B-H data of the particular machine's core material should be used in the FE program to acquire good correlation between the measured and the simulated results.

7.2 Recommendations

The following recommendations are made:

- Further work needs to be done to incorporate the effect of the machine core losses in the simulation of the current waveform under SPMO, and hence to investigate to what extent core losses have an effect.

- To minimise the number of solutions by using larger time or position steps but still maintaining the same accuracy as using Euler integration

with small time or position steps is to impliment more accurate Rugga Kutta integration method in the simulation program.

- More denser mesh has shown to have more accurate solution but longer FE solution. To maintain the same accuracy but with as short FE solution as possible, the mesh used in the FE analysis should be optmimised more especially in the more critical regions in terms of the flux flow. This can be done by mainly focusing at the number of the air gap nodes, which is a more critical region in the SRM.

Appendix A- Brief Mathematical Background On Partial Differentiation and General Euler Integration

A.1. Partial Differentiation

Partial differentiation is a differentiation method to differentiate functions of several variables. A chain rule mathematical theorem composed of partial derivatives is used in the partial differential method of simulation discussed in Chapter 3.

A.1.1 Mathematical Definition

If f is a function of two variables x and y, then the first partial derivative of f with respect to x and y are then functions f_x and f_y denoted by the relations

$$f_x(x,y) = \frac{\partial f}{\partial x} \tag{1}$$

$$f_y(x,y) = \frac{\partial f}{\partial y}, \tag{2}$$

where f_x and f_y are the partial derivatives of f with respect to x and y. To find, f_x we may regard y as a constant and differentiate $f(x,y)$ with respect to x in a usual way. Similarly, to find f_y the variable x is regarded as a constant and $f(x,y)$ is differentiated with respect to variable y. For brevity we speak of $\frac{\partial f}{\partial x}$ or $\frac{\partial f}{\partial y}$ as a partial derivative of f with respect to x or y respectively.

A.1.2 Chain rule mathematical theorem:

If $w = f(u_1,u_2,\cdots,u_n)$ and each u_i, for $i = 1,2,\cdots,n$ is a function of one variable t such that $\frac{du_i}{dt}$ is continuous, then w is a function of t and

$$\frac{dw}{dt} = \frac{\partial w}{\partial u_1}\frac{du_1}{dt} + \frac{\partial w}{\partial u_2}\frac{du_2}{dt} + \cdots + \frac{\partial w}{\partial u_n}\frac{du_n}{dt} \tag{4}$$

This mathematical theorem has been applied in Faraday's voltage equations given in Chapter 3.

A.2. General Euler's Method

Euler's method to "pretend" that the slope of a solution curve remains constant over short intervals is usually not true in general. Suppose that we are given first order differential equation $y' = f(t, y)$ and an initial condition $y(a) = y_0$, and that we use Euler's method with n steps on the interval $a \le t \le b$.

- **What Euler's method produces: graphs and numbers**

What exactly does Euler's method give? The answer depends on our point of view:

Graphically speaking : Euler's method says how to move step by step through the differential equation's slope field, starting at (a, y_0) and ending when $t = b$. Every Euler step has horizontal length $\Delta t = (b - a)/n$, called the step size. Each step goes in the direction determined by the slope field at the beginning point of the step. Joining successive Euler points with line segments produces a piecewise-linear curve. This curve is the graph of an approximate solution function $Y(t)$ for the differential equation as shown in figure A1.

$$\frac{dw}{dt} = \frac{\partial w}{\partial u_1}\frac{du_1}{dt} + \frac{\partial w}{\partial u_2}\frac{du_2}{dt} + \cdots + \frac{\partial w}{\partial u_n}\frac{du_n}{dt} \tag{4}$$

This mathematical theorem has been applied in Faraday's voltage equations given in Chapter 3.

A.2. General Euler's Method

Euler's method to "pretend" that the slope of a solution curve remains constant over short intervals is usually not true in general. Suppose that we are given first order differential equation $y' = f(t, y)$ and an initial condition $y(a) = y_0$, and that we use Euler's method with n steps on the interval $a \le t \le b$.

- **What Euler's method produces: graphs and numbers**

What exactly does Euler's method give? The answer depends on our point of view:

Graphically speaking : Euler's method says how to move step by step through the differential equation's slope field, starting at (a, y_0) and ending when $t = b$. Every Euler step has horizontal length $\Delta t = (b-a)/n$, called the step size. Each step goes in the direction determined by the slope field at the beginning point of the step. Joining successive Euler points with line segments produces a piecewise-linear curve. This curve is the graph of an approximate solution function $Y(t)$ for the differential equation as shown in figure A1.

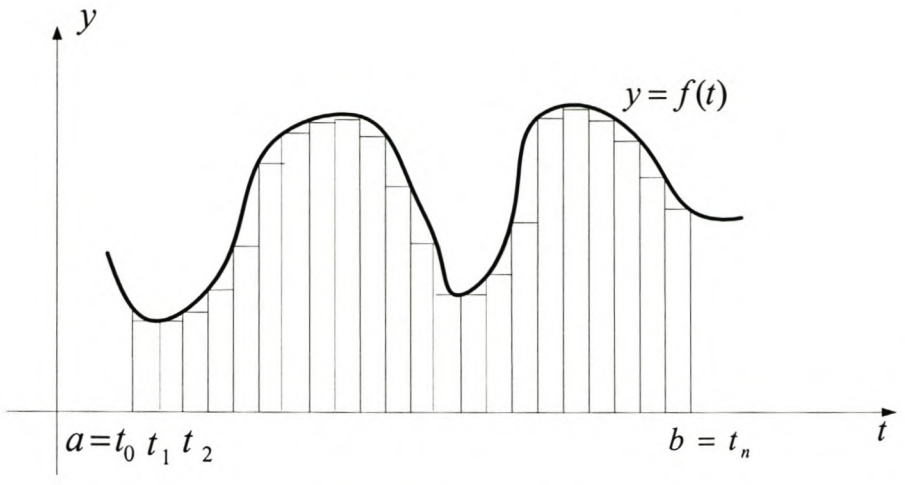


Figure A1 1: Euler's step solution curve

Numerically speaking: Euler's method a list

$$Y(t_0), Y(t_1), Y(t_2), Y(t_3), \ldots, Y(t_n)$$

of numerical values of an approximate numerical solution function Y(t), for equally spaced inputs

$a = t_0 < t_1 < t_2 < t_3 < \cdots < t_n = b$;

each t_i is Δt units to the right of its predecessor. (Plotting these values and joining successive points with line segments produces the graph mentioned above.)

- **What Euler's method doesn't produce: formulas**

Euler's method uses-and produces-only numerical information. We can use Euler's method, in other words, to produce a graph or table of values, but not a "formula" for a solution function. To its credit, Euler's method can be used with almost any differential equation, no matter how complicated the equation is. To its debit, Euler's method produces only approximate results, even for simple DE's that might otherwise be solved exactly.

- **Euler's method step by step**

How, in general, does Euler's method produce the successive values

$Y(t_0), Y(t_1), Y(t_2), Y(t_3), \ldots, Y(t_n)$?

Step 1: The starting point $Y(t_0) = y_0$ comes from the initial condition. By the differential equation, a solution curve through (t_0, y_0) has slope $m_0 = f(t_0, y_0)$, so the first Euler step has slope m_0. Its rise (or fall), therefore, is $m_0 . \Delta t$ vertical units:

$$Y(t_1) = Y(t_0) + m_0 \Delta t$$

Step 2: The second Euler step starts at $(t_1, Y(t_1))$. At $(t_1, Y(t_1))$, by the differential equation (DE), a solution curve has slope $m_1 = f(t_1, Y(t_1))$. Thus the second Euler step rises (or falls) $m_1 . \Delta t$ vertical units:

$$Y(t_2) = Y(t_1) + m_1 \Delta t$$

Continue: The pattern should now be clear. At each step we (1) use the DE to "update" the slope m_i; (2) move with slope m_i from $(t_i, Y(t_i))$ to $(t_{i+1}, Y(t_{i+1}))$. After n steps we arrive at $t_n = b$. Reduced entirely to symbols, Euler's recipe for assembling an approximate solution $Y(t)$ looks like this:

$$\begin{aligned}
Y(t_0) &= y_0; \\
Y(t_1) &= Y(t_0) + f(t_0, Y(t_0)) . \Delta t; \\
Y(t_2) &= Y(t_1) + f(t_1, Y(t_1)) . \Delta t; \\
&\vdots \\
Y(t_{i+1}) &= Y(t_0) + f(t_0, Y(t_0)) . \Delta t; \\
&\vdots \\
Y(t_n) &= Y(t_{n-1}) + f(t_{n-1}, Y(t_{n-1})) . \Delta t .
\end{aligned}$$

Appendix B-Finite Element Program of the SRM

```
c*******************************************************
c Program for the mesh generation
c*******************************************************
c ===========================================================
c  Rotor mesh
c ===========================================================
c
c  Sub rotor_fbr generates the poly-shape for a switched reluctance
c  rotor with four poles.
      subroutine rotor_fbr(nseg_pol,nspnts_pol,nstype_pol,x_pol,
     &   y_pol,ro_od,ro_shfd,ro_tw,ro_th,ro_flr,ro_yh,bw_nw,an_nw,
     &   i_t1,i_t2,i_t3)
c
c  Variable declaration
            implicit double precision (a-h,o-z)
            implicit integer (i-n)
            double precision pi
            double precision sqrt2,dg_rad
      integer i,j,i_t1,i_t2,i_t3
c poly-shape variables:
            integer nseg_pol
            integer nspnts_pol(1101),nstype_pol(1101)
            double precision x_pol,y_pol,arc,an_nw,bw_nw
     dimension x_pol(1101,1101),y_pol(1101,1101),arc(100)
            dimension an_nw(4),bw_nw(4)
c
c rotor variables:
            double precision ro_od,ro_shfd,ro_tw,
     + ro_th,ro_flr,ro_yh,magnh

c arc variables:
            double precision ainc,arcrad,
     + arc_x,arc_y,angl,delx,dely,gen
            double precision arc1,arc2,arc3,arc4,arc5,arc6,arc7,arc8,
     + x_30,y_30,x_40,y_40,x_44,y_44,x_44_7,y_44_7
            pi=2.0d0*dasin(1.0d0)
            sqrt2=dsqrt(2.0d0)
            dg_rad=pi/180.0d0
            magnh=0.002d0

c Start building the rotor-pole mesh:
c ----------------------------------
c
            nseg_pol=44

c Enter type for polygons including magnet material.
c Type and direction for magnet coil (sheet) polygons
c are always given directly in ee_pol: for rotor = 4.
```

```
c In this design the numbers are 37 and 43. Remember to
c check the mirror magnet coil directions at the end!

      do i=1,2
       nstype_pol(i)=0
      end do
      do i=3,22
       nstype_pol(i)=250
      end do
      do i=23,44
       nstype_pol(i)=0
       if (i.eq.37) nstype_pol(i)= 0
       if (i.eq.43) nstype_pol(i)= 0
       if (i.eq.38) nstype_pol(i)= 0
       if (i.eq.39) nstype_pol(i)= 0
       if (i.eq.41) nstype_pol(i)= 0
       if (i.eq.42) nstype_pol(i)= 0
      end do


c number of points for polygons

      nspnts_pol(1)=6
            nspnts_pol(2)=10
      nspnts_pol(3)=13
      nspnts_pol(4)=13
      nspnts_pol(5)=19
      nspnts_pol(6)=19
      nspnts_pol(7)=16
      nspnts_pol(8)=15
      nspnts_pol(9)=15
      nspnts_pol(10)=17
      nspnts_pol(11)=16
            nspnts_pol(12)=16
            nspnts_pol(13)=16
            nspnts_pol(14)=18
      nspnts_pol(15)=12
      nspnts_pol(16)=12
      nspnts_pol(17)=13
      nspnts_pol(18)=14
      nspnts_pol(19)=18
      nspnts_pol(20)=18
      nspnts_pol(21)=18
      nspnts_pol(22)=19
      nspnts_pol(23)=15
      nspnts_pol(24)=16
      nspnts_pol(25)=14
        nspnts_pol(26)=16
      nspnts_pol(27)=9
            nspnts_pol(28)=10
      nspnts_pol(29)=9
```

```
      nspnts_pol(30)=12
      nspnts_pol(31)=10
      nspnts_pol(32)=10
            nspnts_pol(33)=9
            nspnts_pol(34)=11
            nspnts_pol(35)=12
            nspnts_pol(36)=10
            nspnts_pol(37)=5
            nspnts_pol(38)=15
            nspnts_pol(39)=15
            nspnts_pol(40)=11
            nspnts_pol(41)=10
            nspnts_pol(42)=9
            nspnts_pol(43)=7
            nspnts_pol(44)=11

c calculate angles for num points
c on airgap arc i.e. half pole pitch:
            num=56
      ainc=pi/4.0d0/dble(num)
      angl=ainc
      do i=1,num
       arc(i)=pi/4.0d0+angl
       angl=angl+ainc
      end do

c calculate arcs

      arc1=ro_shfd/4.0d0
      arc2=ro_shfd/2.0d0
      arc3=(ro_shfd+ro_yh)/2.0d0
      arc4=ro_shfd/2.0d0+ro_yh
      arc5=arc4+2.0d0*ro_th/6.0d0
      arc6=arc4+4.0d0*ro_th/6.0d0
      arc7=arc4+5.0d0*ro_th/6.0d0
      arc8=arc4+ro_th


c determine the points of the rotor
c -------------------------------------


c points for region 1

      x_pol(1,1)= 0.0d0
      y_pol(1,1)= 0.0d0
      x_pol(1,3)= arc1
      y_pol(1,3)= 0.0d0
      x_pol(1,2)= (x_pol(1,1)+x_pol(1,3))/2.0d0
      y_pol(1,2)= 0.0d0
      arcang= pi/8.0d0
```

```
      call dpolrec(arc1,arcang,arc_x,arc_y)
      x_pol(1,4)= arc_x
      y_pol(1,4)= arc_y
      arcang= pi/4.0d0
      call dpolrec(arc1,arcang,arc_x,arc_y)
      x_pol(1,5)= arc_x
      y_pol(1,5)= arc_y
      x_pol(1,6)= x_pol(1,5)/2.0d0
            y_pol(1,6)= y_pol(1,5)/2.0d0

c points for region 2

      x_pol(2,1)= x_pol(1,3)
      y_pol(2,1)= y_pol(1,3)
      x_pol(2,3)= arc2
      y_pol(2,3)= 0.0d0
      x_pol(2,2)= (x_pol(2,1)+x_pol(2,3))/2.0d0
      y_pol(2,2)= 0.0d0
      x_pol(2,9)= x_pol(1,5)
      y_pol(2,9)= y_pol(1,5)
      x_pol(2,10)= x_pol(1,4)
      y_pol(2,10)= y_pol(1,4)
      arcang= pi/16.0d0
            do i=4,7
              angl= arcang*(i-3)
       call dpolrec(arc2,angl,arc_x,arc_y)
       x_pol(2,i)= arc_x
       y_pol(2,i)= arc_y
            end do
     x_pol(2,8)= (x_pol(2,7)+x_pol(2,9))/2.0d0
     y_pol(2,8)= (y_pol(2,7)+y_pol(2,9))/2.0d0
                        .
                        .
                        .
c Up to the last regions

c points for region 43

            x_pol(43,1)= x_pol(42,4)
            y_pol(43,1)= y_pol(42,4)
            x_pol(43,2)= x_pol(22,13)
            y_pol(43,2)= y_pol(22,13)
            x_pol(43,5)= x_pol(26,5)
            y_pol(43,5)= y_pol(26,5)
            x_pol(43,6)= x_pol(42,6)
            y_pol(43,6)= y_pol(42,6)
            x_pol(43,7)= x_pol(42,5)
            y_pol(43,7)= y_pol(42,5)
            arc_x=x_pol(43,5)
            arc_y=y_pol(43,5)
```

```
            call drecpol(arc_x,arc_y,gen,arc(5))
          arcang= (arc(5)-arc(4))/3.0d0
            do i=3,4
              angl= arc(4)+arcang*dble(i-2)
           call dpolrec(arc8,angl,arc_x,arc_y)
           x_pol(43,i)= arc_x
           y_pol(43,i)= arc_y
         end do

c points for region 44

          x_pol(44,1)= x_pol(40,9)
               y_pol(44,1)= y_pol(40,9)
          do i= 9,11
            x_pol(44,i)= x_pol(36,(15-i))
            y_pol(44,i)= y_pol(36,(15-i))
          end do
          gen= arc8/arc8
          delx = gen*(x_44_7-x_pol(44,9))/2.0d0
          dely = gen*(y_44_7-y_pol(44,9))/2.0d0
          do i= 1,2
           x_pol(44,(9-i))= x_pol(44,9)+i*delx
           y_pol(44,(9-i))= y_pol(44,9)+i*dely
          end do
               x_pol(44,2)= x_pol(40,8)
               y_pol(44,2)= y_pol(40,8)
          x_pol(44,3)= x_pol(40,7)
               y_pol(44,3)= y_pol(40,7)
               gen= (y_pol(44,7)-y_44)/4.0d0
          do i=4,6
               y_pol(44,i)=y_44+(i-3)*gen
          x_pol(44,i)=((arc8)**2.0d0-(y_pol(44,i))**2.0d0)**0.5d0
               end do

c quarter regions have been defined
c turn quarter of a pole into half a pole
c remember magnet coil directions (polygons 37 and 43)

             do i= 1,nseg_pol
               k=nseg_pol+i
               nspnts_pol(k)=nspnts_pol(i)
         nstype_pol(k)=nstype_pol(i)
         do j=1,nspnts_pol(i)
          l=nspnts_pol(i)+1-j
          x_pol(k,l)= 1.0d0*y_pol(i,j)
          y_pol(k,l)= x_pol(i,j)
         end do
            end do
        nstype_pol(37+nseg_pol)=-nstype_pol(37)
        nstype_pol(43+nseg_pol)=-nstype_pol(43)
```

```
            nseg_pol=nseg_pol*2

c half pole regions have been defined
c turn half a pole into a full pole
c
            do i=1,nseg_pol
              k=nseg_pol+i
              nspnts_pol(k)=nspnts_pol(i)
       nstype_pol(k)=nstype_pol(i)
       do j=1,nspnts_pol(i)
        l=nspnts_pol(i)+1-j
        x_pol(k,l)=-1.0d0*x_pol(i,j)
        y_pol(k,l)= y_pol(i,j)
       end do
            end do
      nstype_pol(37+nseg_pol)=-nstype_pol(37)
      nstype_pol(43+nseg_pol)=-nstype_pol(43)
        nstype_pol(37+(3*nseg_pol/2))=nstype_pol(37)
      nstype_pol(43+(3*nseg_pol/2))=nstype_pol(43)
            nseg_pol=nseg_pol*2

c finished

            return
            end
c end of sub rotor_fbr
c --------------------

c ======================================================
c  Stator mesh
c ======================================================

            subroutine stator_slot_ss(sl_type,spitch,
     + nseg_pol,nspnts_pol,nstype_pol,x_pol,y_pol,
     + n_st_st,st_od,st_id,st_gw,st_bw,st_wid,st_waw,
     + st_sht,st_shs,st_shb,st_wbw,st_yh,st_wag,st_twg,
     + st_ft,st_f1,st_f2,st_f3,i_t1,i_t2,i_t3,ii)


c----------------------
c  Variable declaration
            implicit double precision (a-h,o-z)
            implicit integer (i-n)
            double precision pi
            integer i
c
c poly-shape variables:
            character*3 sl_type
            integer nseg_pol
            integer nspnts_pol(1101),nstype_pol(1101)
```

```
            double precision x_pol,y_pol
      dimension x_pol(1101,1101),y_pol(1101,1101)
c
c stator variables:
            integer n_st_st
      double precision st_od,st_id,st_gw,st_bw,
     + st_wid,st_waw,st_wag,st_twg,st_ft,st_f1,st_f2,st_f3,
     + st_sht,magnh1,magnh2,magnh3,st_yh,st_wbw,
     + st_shs,st_shb,spitch,gap_w,rib_h,rr
c
c winding

            integer i_t1,i_t2,i_t3

c
c arc variables:
            double precision ap1,ap2,ainc,ainc1,arcrad,arcang,
     + arc_x,arc_y,x_39_5,y_39_5,x_60_8,y_60_8,
     + x_5_3,y_5_3,x_1_1,y_1_1,arcinc,webh,shitht,R_29_ht,gen

c
c to determine slot-fill points:
            double precision rr
            integer itd

      R_29_ht=0.011510529
      magnh1=0.002d0
      magnh2=0.002d0
      magnh3=0.001d0
      gap_w=0.002d0
      rib_h=0.002d0
      webh=0.002d0
      shitht=0.002d0

            pi=2.0d0*dasin(1.0d0)
            spitch=2.0d0*pi/dble(n_st_st)
            itd=0

c
c Start building the stator-slot mesh:
c
      nseg_pol=60
c
c the number of points per segment, and types:
c sections 1 to 12 does the slot
c sections 13 to 24 covers the lamination around the slot
c
c Region 1: slot-gap opening at bottom far-right
            nspnts_pol(1)=16
            nstype_pol(1)=0
```

```
c Region 2: slot-gap opening at bottom-far right
            nspnts_pol(2)=16
            nstype_pol(2)=0
c Region 3: slot-gap opening at bottom-far left
      nspnts_pol(3)=16
            nstype_pol(3)=0
c Region 4: slot-gap opening at bottom left
      nspnts_pol(4)=12
            nstype_pol(4)=0
c Region 5: top of Section 1 and 3
            nspnts_pol(5)=7
            nstype_pol(5)=0
c Region 6: top of section 2 and 4
            nspnts_pol(6)=10
            nstype_pol(6)=0
c R 7: right of region 5
            nspnts_pol(7)=6
            nstype_pol(7)=0
c R 8: left of region 6
            nspnts_pol(8)=12
            nstype_pol(8)=i_t2
c R 9: top of region 5
            nspnts_pol(9)=16
            nstype_pol(9)=i_t2
c R10: top of region 6
            nspnts_pol(10)=16
            nstype_pol(10)=i_t2
c R11: top of region 7
            nspnts_pol(11)=12
            nstype_pol(11)=i_t2
c R12: top of region 8
            nspnts_pol(12)=12
            nstype_pol(12)=i_t2
c R13: left of slot-gap opening
      nspnts_pol(13)=10
            nstype_pol(13)=0
c R14: right of slot-gap opening
      nspnts_pol(14)=6
            nstype_pol(14)=0
c R15: far left of slot-gap opening
      nspnts_pol(15)=11
            nstype_pol(15)=0
c R16: far right of slot-gap opening
      nspnts_pol(16)=18
            nstype_pol(16)=i_t2
c R17: top of region 13 and 15
            nspnts_pol(17)=17
            nstype_pol(17)=i_t2
c R18: top of region 14 and 16
            nspnts_pol(18)=11
```

```
            nstype_pol(18)=i_t2
c R19: top of region 17
            nspnts_pol(19)=12
            nstype_pol(19)=i_t2
c R20: top of region 18
            nspnts_pol(20)=14
            nstype_pol(20)=i_t2
c R21: top left-middle
            nspnts_pol(21)=10
            nstype_pol(21)=i_t2
c R22: top right-middle
            nspnts_pol(22)=14
            nstype_pol(22)=i_t2
c R23: far top left
            nspnts_pol(23)=16
            nstype_pol(23)=i_t2
c R24: far top right
            nspnts_pol(24)=13
            nstype_pol(24)=i_t2
c R25: top of region 13 and 15
            nspnts_pol(25)=6
            nstype_pol(25)=i_t2
c R26: top of region 14 and 16
            nspnts_pol(26)=7
            nstype_pol(26)=201
c R27: top of region 17
            nspnts_pol(27)=6
            nstype_pol(27)=201
c R28: top of region 18
            nspnts_pol(28)=11
            nstype_pol(28)=201
c R29: top left-middle
            nspnts_pol(29)=11
            nstype_pol(29)=201
c R30: top right-middle
            nspnts_pol(30)=6
            nstype_pol(30)=201
c For magnet in yoke change regions 26-31
c only at one place:
            if (ii.eq.2) then
              nstype_pol(26)=201
c             nstype_pol(27)=4
              nstype_pol(27)=201
c             nstype_pol(28)=12
              nstype_pol(28)=201
c             nstype_pol(29)=12
              nstype_pol(29)=201
c             nstype_pol(30)=-4
              nstype_pol(30)=201
              nstype_pol(31)=201
```

```
      arc10=magnh1
      arcinc= (arc2-arc10)/4.0d0

c calculate the stator arcs

       arc5=st_id/2.0d0
       arc2=arc_2
       arc1=0.5d0*arc2
       arc4=arc_2+st_yh
       arc3=(arc4+arc2)/2.0d0
       arc6=0.5d0*arc1
       arc7=(arc1+arc2)/2.0d0
             arc8=0.5d0*arc6
             arc9=0.5d0*arc8

c calculate the stator fixed points

      arcrad= arc5
            arcang= pi/3.0d0
            call dpolrec(arcrad,arcang,arc_x,arc_y)
            x_39_5= arc_x
            y_39_5= arc_y

         arcrad= arc5+arc4
            arcang= pi/3.0d0
            call dpolrec(arcrad,arcang,arc_x,arc_y)
            x_60_8= arc_x
            y_60_8= arc_y

      x_30_6= 0.0d0
      y_30_6= arc5+arc4

      arcrad=arc5+arc4
      call drecpol( x_60_8,y_60_8,rr,ap2)
            call drecpol( x_pol(35,8),y_pol(35,8),rr,ap1)
            ainc1=(ap1-ap2)/3.0d0

            ap1=ap1-ainc1
      call dpolrec(arcrad,ap1,arc_x,arc_y)
      x_58_8= arc_x
            y_58_8= arc_y
            ap1=ap1-ainc1
      call dpolrec(arcrad,ap1,arc_x,arc_y)
      x_59_8= arc_x
            y_59_8= arc_y
      arcrad= arc5+arc4
            arcang= pi/3.0d0
            call dpolrec(arcrad,arcang,arc_x,arc_y)
            x_60_8= arc_x
            y_60_8= arc_y
```

```
c calculate angles

      x_5_3= st_wid/2.0d0
      y_5_3= ((arc5)**2.0d0-(x_5_3)**2.0d0)**0.5d0
      x_4_3= x_5_3-magnh2
      y_4_3= ((arc5)**2.0d0-(x_4_3)**2.0d0)**0.5d0
      x_1_1= 0.0d0
      y_1_1= arc5
      call drecpol( x_1_1,y_1_1,rr,ap1)
            call drecpol( x_4_3,y_4_3,rr,ap2)
            ainc=(ap1-ap2)/14.0d0

c points for region 1

      x_pol(1,1)= 0.0d0
        y_pol(1,1)= y_1_1

      do i=2,5
               call dpolrec(arc5,(ap1-ainc*dble(i-1)),arc_x,arc_y)
               x_pol(1,i)=arc_x
               y_pol(1,i)=arc_y
      end do
      arcrad=arc5+arc8
      x_pol(1,13)= 0.0d0
      y_pol(1,13)= arcrad
      x_pol(1,12)= gap_w
      y_pol(1,12)= ((arcrad)**2.0d0-(x_pol(1,12))**2.0d0)**0.5d0

      x_pol(1,9)= x_pol(1,5)
      y_pol(1,9)= ((arcrad)**2.0d0-(x_pol(1,9))**2.0d0)**0.5d0
      call drecpol( x_pol(1,9),y_pol(1,9),rr,ap2)
            call drecpol( x_pol(1,12),y_pol(1,12),rr,ap1)
            ainc=(ap1-ap2)/3.0d0
      do i=2,4
               call dpolrec(arcrad,(ap1-ainc*dble(i-1)),arc_x,arc_y)
               x_pol(1,(13-i))=arc_x
               y_pol(1,(13-i))=arc_y
            end do
      arcrad= arc5+magnh1
      x_pol(1,7)= x_pol(1,5)
      y_pol(1,7)= ((arcrad)**2.0d0-(x_pol(1,7))**2.0d0)**0.5d0
      y_pol(1,6)= (y_pol(1,5)+y_pol(1,7))/2.0d0
      x_pol(1,6)= x_pol(1,5)
      y_pol(1,8)= (y_pol(1,9)+y_pol(1,7))/2.0d0
      x_pol(1,8)= x_pol(1,5)
      arcrad= arc5+magnh1
      x_pol(1,15)= x_pol(1,1)
      y_pol(1,15)= ((arcrad)**2.0d0-(x_pol(1,15))**2.0d0)**0.5d0
```

```
      y_pol(1,14)= (y_pol(1,13)+y_pol(1,15))/2.0d0
      x_pol(1,14)= x_pol(1,1)
      y_pol(1,16)= (y_pol(1,15)+y_pol(1,1))/2.0d0
      x_pol(1,16)= x_pol(1,1)

c points for region 2

      x_pol(2,1)= x_pol(1,5)
      y_pol(2,1)= y_pol(1,5)
            call drecpol( x_pol(2,1),y_pol(2,1),rr,ap1)
      do i=2,5
            call dpolrec(arc5,(ap1-ainc*dble(i-1)),arc_x,arc_y)
            x_pol(2,i)=arc_x
            y_pol(2,i)=arc_y
            end do
      do i= 1,4
        x_pol(2,(12+i))= x_pol(1,(10-i))
        y_pol(2,(12+i))= y_pol(1,(10-i))
      end do
      arcrad=arc5+arc8
      x_pol(2,9)= x_pol(2,5)
            y_pol(2,9)= ((arcrad)**2.0d0-(x_pol(2,9))**2.0d0)**0.5d0
      call drecpol( x_pol(2,9),y_pol(2,9),rr,ap2)
            call drecpol( x_pol(2,13),y_pol(2,13),rr,ap1)
            ainc=(ap1-ap2)/4.0d0
      do i=2,4
              call dpolrec(arcrad,(ap1-ainc*dble(i-1)),arc_x,arc_y)
              x_pol(2,(14-i))=arc_x
              y_pol(2,(14-i))=arc_y
            end do
      arcrad= arc5+magnh1
      x_pol(2,7)= x_pol(2,5)
      y_pol(2,7)= ((arcrad)**2.0d0-(x_pol(2,7))**2.0d0)**0.5d0
      y_pol(2,6)= (y_pol(2,5)+y_pol(2,7))/2.0d0
      x_pol(2,6)= x_pol(2,5)
      y_pol(2,8)= (y_pol(2,9)+y_pol(2,7))/2.0d0
      x_pol(2,8)= x_pol(2,5)
                  .
                  .
                  .
c Up to the last regions

c points for region 59

      do i= 1,4
        x_pol(59,(i+10))= x_pol(58,(9-i))
        y_pol(59,(i+10))= y_pol(58,(9-i))
      end do
      do i= 1,4
        x_pol(59,i)= x_pol(56,(12-i))
```

```
      y_pol(59,i)= y_pol(56,(12-i))
    end do

    call drecpol( x_pol(59,11),y_pol(59,11),rr,ap1)
          gen=(ap1-pi/3.0d0)/6.0d0
    arcrad= arc5+arc4
    do i=2,4
          arcang= ap1-(i-1)*gen
     call dpolrec(arcrad,arcang,arc_x,arc_y)
     x_pol(59,(12-i))= arc_x
     y_pol(59,(12-i))= arc_y
          end do
    x_pol(59,6)= (x_pol(59,8)+x_pol(59,4))/2.0d0
    y_pol(59,6)= (y_pol(59,8)+y_pol(59,4))/2.0d0
    x_pol(59,5)= (x_pol(59,6)+x_pol(59,4))/2.0d0
    y_pol(59,5)= (y_pol(59,6)+y_pol(59,4))/2.0d0
    x_pol(59,7)= (x_pol(59,8)+x_pol(59,6))/2.0d0
    y_pol(59,7)= (y_pol(59,8)+y_pol(59,6))/2.0d0

c points for region 60

    do i= 1,4
      x_pol(60,(i+10))= x_pol(59,(9-i))
      y_pol(60,(i+10))= y_pol(59,(9-i))
    end do
    do i= 1,4
      x_pol(60,i)= x_pol(57,(12-i))
      y_pol(60,i)= y_pol(57,(12-i))
    end do
 call drecpol( x_pol(60,11),y_pol(60,11),rr,ap1)
          gen=(ap1-pi/3.0d0)/3.0d0
    arcrad= arc5+arc4
    do i=2,4
          arcang= ap1-(i-1)*gen
     call dpolrec(arcrad,arcang,arc_x,arc_y)
     x_pol(60,(12-i))= arc_x
     y_pol(60,(12-i))= arc_y
          end do
    x_pol(60,6)= (x_pol(60,8)+x_pol(60,4))/2.0d0
    y_pol(60,6)= (y_pol(60,8)+y_pol(60,4))/2.0d0
    x_pol(60,5)= (x_pol(60,6)+x_pol(60,4))/2.0d0
    y_pol(60,5)= (y_pol(60,6)+y_pol(60,4))/2.0d0
    x_pol(60,7)= (x_pol(60,8)+x_pol(60,6))/2.0d0
    y_pol(60,7)= (y_pol(60,8)+y_pol(60,6))/2.0d0

c regions have been defined
c turn quoter of a pole into half a pole
c and remember to change manual the winding types.
          do i= 1,nseg_pol
           k=nseg_pol+i
```

```
            nspnts_pol(k)=nspnts_pol(i)
          nstype_pol(k)=nstype_pol(i)
          do j=1,nspnts_pol(i)
           l=nspnts_pol(i)+1-j
           x_pol(k,l)= -1.0d0*x_pol(i,j)
           y_pol(k,l)= y_pol(i,j)
          end do
              end do
        nstype_pol(nseg_pol+8)=i_t1
        nstype_pol(nseg_pol+9)=i_t1
        nstype_pol(nseg_pol+10)=i_t1
        nstype_pol(nseg_pol+11)=i_t1
        nstype_pol(nseg_pol+12)=i_t1
        nstype_pol(nseg_pol+16)=i_t1
        nstype_pol(nseg_pol+17)=i_t1
        nstype_pol(nseg_pol+18)=i_t1
        nstype_pol(nseg_pol+19)=i_t1
        nstype_pol(nseg_pol+20)=i_t1
        nstype_pol(nseg_pol+21)=i_t1
        nstype_pol(nseg_pol+22)=i_t1
        nstype_pol(nseg_pol+23)=i_t1
        nstype_pol(nseg_pol+24)=i_t1
        nstype_pol(nseg_pol+25)=i_t1
        nseg_pol=nseg_pol*2

c end do nseg reflected
              return
              end
```

Unaligned, inbetween and completely aligned positions

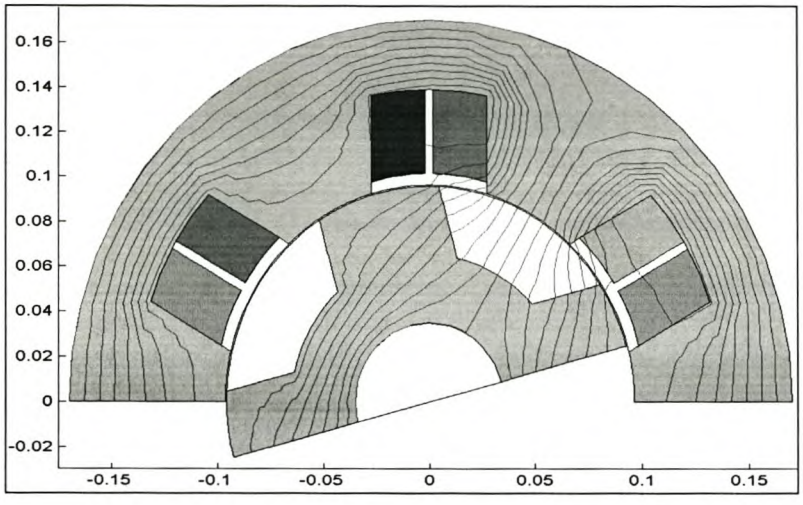


Figure B1 1: Finite element field solution of the tapered stator pole SRM at the-unaligned 0 degrees position

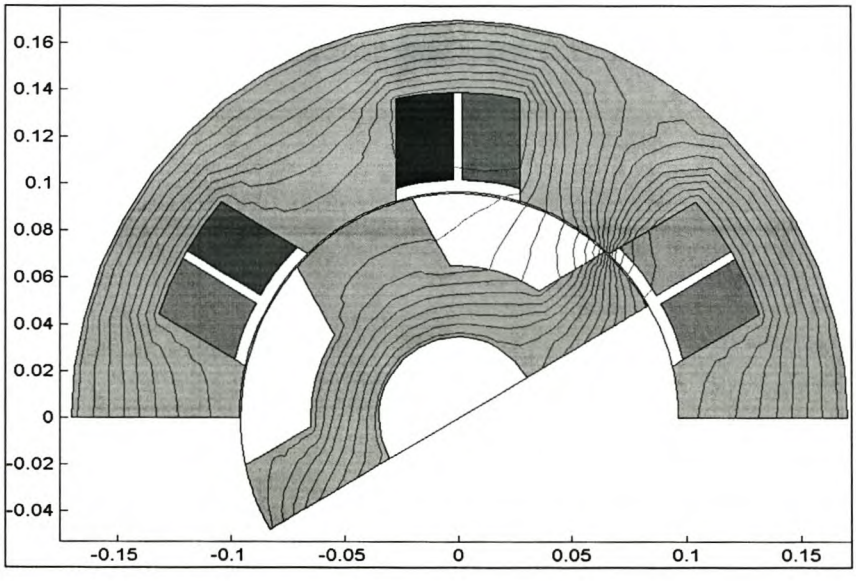


Figure B1 2: Finite element field solution of the tapered stator pole SRM at 15 degree position

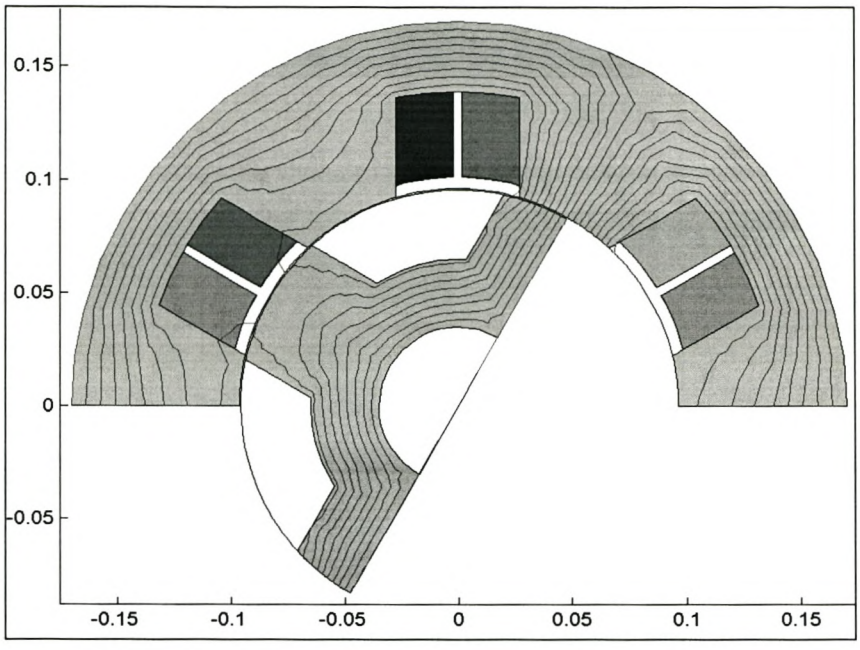


Figure B1 3: Finite element field solution of the tapered stator pole SRM at 45 degree position

```
c ******************************************************
c Program for the polynomial method of simulation
c ******************************************************

c variable declarations

      parameter(ndim=2,tol=5.0d-5,PI=3.141593)

      integer m,time,N,k,l,h,jk,w
      double precision xpar(50),ypar(50),xk(1000),th(250),V(250)
      double precision pvar(10),po(10),xi(ndim,ndim),pe(10),t(250),
     &   f_link(250),cur(250),fk(1000)
      double precision xn,fn,fo,fe,fmp,del,dim(2,ndim),del_t,t0,t_fin,
     &   del_th,del_x,ro_po(250),fnn
      double precision avdir(ndim),a,b,p_old(10),tyd,cur_fin,eps
      real R, speed,dth1,pdth,off,th_f,th_I


      open (33,file="results/perform.res", form='formatted')
      open (44,file="results/tperform.res",form='formatted')
      rewind 33
      rewind 44
      write(33,100)
      write(44,100)
  100  format('')
  131  format(10F9.3)
  133  format('Optimum',2I3,6F9.3)

      write(33,151)
  151  format('Start 5/4/02 Simulation: SRM')

            print *,' '
            print *,'--> Start with SRM simulation'
      nsol=nsol+1


            xpar(2)=-15.0d0
      cur_fin=0.0d0
            fn=0.0d0
      write(33,131) xpar(2)+15.0d0,cur_fin,fnn
      del=(dim(2,1)-dim(1,1))/idiv
      ik=1
      jk=1
c Start with outer loop, i.e. integration
      R=0.3089
      V(1)=500.0d0
      speed=50.0d0*PI
      t_fin=(PI/3.0d0)/speed
      t0=0.0d0
```

```
      N=60
      del_t=(t_fin-t0)/N
      t(1)=0.0d0
      th(1)=0.0d0
      dth1=PI/4.0d0
      pdth=(0.3333)*dth1
      off=0.666666
      do k=2,N
       t(k)=t(1)+(k-1)*del_t
       th(k)=speed*t(k)
       ro_po(k)=(180.0d0/PI)*th(k)-15.0d0

       if (th(k).le.pdth) then
        V(k)=500.0d0
       else if (th(k).gt.pdth.and.th(k).le.off*dth1) then
        V(k)=500.0d0
       else if (th(k).gt.off*dth1.and.th(k).le.dth1) then
        V(k)=-500.0d0
       else if (th(k).gt.dth1) then
        V(k)=-500.0d0
       end if

      end do
       f_link(1)=0.0d0
       xpar(1)=0.0d0
       cur(1)=0.0d0
       xk(1)=xpar(1)
       do j=2,1000
          xk(j)=xk(1)+(j-1)*del
       end do

       do h=2,N
        eps=V(h-1)-R*cur(h-1)
        f_link(h)=f_link(h-1)+eps*del_t
        fnn=f_link(h)
        xpar(2)=ro_po(h)
              print*,'ro_po(h)=',ro_po(h)
              print *,' '
              print *,'Step, Flux linkage, Position =',h,fnn,xpar(2)
              if (V(h-1).eq.0.0d0) then
               cur_fin=0.0d0
        cur(h)=cur_fin
        go to 33
       end if
c Start with inner loop, i.e. solve for new current

  1  do i=ik,ik+2
     xpar(1)=xk(i)
     CALL eesolv(xpar,ypar)
```

```
      fk(i)= ypar(1)
            print *,'Flux linkage, Current = ',ypar(1),xpar(1)
     end do

            if (fnn.lt.0.0d0) then
       cur_fin=0.0d0
      else if (fnn.ge.fk(ik).and.fnn.le.fk(ik+2)) then
              print *,'Call'
              CALL newtonsol(ik,xk,fk,fnn,cur_fin)
              print *,'fnn,cur_fin=',fnn,cur_fin
       else if (fnn.lt.fk(ik)) then
       ik=ik-1
       go to 1
      else if (fnn.gt.fk(ik+2)) then
       ik=ik+1
       go to 1
      end if

     print *,'Phase Current solution = ', cur_fin
     cur(h)=cur_fin
     ik=ik
 33  write(33,131) xpar(2)+15.0d0,cur_fin,fnn
    end do

              print *,'Another run required: decrement ik'
       CALL eesolv(xpar,ypar)
c       print *,'backfromsolve'
       write(33,131) ypar(4),xpar(1),xpar(2),xpar(3),xpar(4),xpar(5),
     +                  xpar(6),xpar(7),xpar(8),xpar(9)
              fk(1)=ypar(4)
              go to 3
             else if (fk(3).gt.fk(2)) then
              ik=ik+1
              if (ik.gt.idiv-2) then
               do j=1,idiv-1

       CALL eesolv(xpar,ypar)
       print *,'backfromsolve after increment ik'
       write(33,131) ypar(4),xpar(1),xpar(2),xpar(3),xpar(4),xpar(5),
     +                  xpar(6),xpar(7),xpar(8),xpar(9)
              fk(3)=ypar(4)
              go to 3
             end if
c end of test
c ------------

c get maximum with Newton:
              fmp=fn
              call newton(ik,xk,fk,xn,fn)
```

```
c save point for next direction:
            do j=1,ndim
             pvar(j)=pvar(j)+xn*xi(j,kdir)
             xpar(j)=pvar(j)
            end do
c ************************************************************
            subroutine newton(ik,xk,fk,xn,fn)

c Use interpolating polinomial of the newton form: second degree.
c Inputs are 3 variables of 'xk' and 3 function values 'fk'.
c Outputs are maximum of curve 'fn' at position 'xn'.
c ************************************************************

            double precision xk(30),fk(3),const(3)
            double precision xn,fn,a

c calculate constants

            const(1)=fk(1)
            const(2)=(fk(1)-fk(2))/(xk(ik)-xk(ik+1))
            a=(fk(2)-fk(3))/(xk(ik+1)-xk(ik+2))
            const(3)=(const(2)-a)/(xk(ik)-xk(ik+2))

            xn=0.5d0*(xk(ik+1)+xk(ik)-const(2)/const(3))
            a=const(3)*(xn-xk(ik))*(xn-xk(ik+1))
            fn=const(1)+const(2)*(xn-xk(ik))+a

            return
            end
c end of subroutine newton
c ------------------------

c ************************************************************
c Partial differential method of simulation
c ************************************************************
c This program simulate single current pulse mode
c operation of a switched reluctance machine (SRM)
c with two phases active at a time using the
c Partial Differential Method of Simulation
c Actual system's inertia is used
c ------------------------------------------------------

            program eeoptmb

c variable declarations

      parameter(PI=3.141593)
```

```
      integer N,k
      double precision xpar(50),ypar(50),th(250),t(250)
      double precision thet(50),va(200),vb(200),vc(200),deli,delthet
      double precision del_t,t0,speed,cura,curb,curc,r_s,t_fin
      double precision fiaa,fibb,ficc,fiaaa,fibbb,ficcc
      double precision duac,ducc,duaa,duca,consa,consc,torq1
      double precision dubb,duab,duba,consb
      double precision ducc,dubc,ducb,bnew
      double precision a1,a2,anew,cnew,delspeed,Tld,Tm,J
      double precision oldspeed,newposition,newspeed,Tnew
      double precision delposition,oldposition

c ---------------------------------------------------------------
c Start SRM simulation program for 3 phase operation
c ---------------------------------------------------------------


      open (33,file="results/perform.res", form='formatted')
      open (44,file="results/tperform.res",form='formatted')
      rewind 33
      rewind 44
      write(33,100)
      write(44,100)
  100 format(10F9.3)
  131 format(10F9.3)
  133 format('Optimum',2I3,6F9.3)

c start simulation
      write(33,151)
  151 format('Start 15/05/03 Simulation: SRM')
      print *,'--> Start with SRM simulation'

c initialize
      r_s = 0.308
      N = 91
        deli = 2.0d0
      cura = 0.0d0
      curb = 0.0d0
      curc = 83.316d0
      delthet = 3.0d0
      oldspeed = 52.4d0*PI
      del_t = 0.00011111111111111111d0
      J = 0.8d0
      Tld = 140.0d0
      delspeed = 0.0d0
      oldposition = 0.0d0
      newposition = oldposition
c ----------------
c Case 1: Phases A and C are active: 0 - 30 degrees
c Phase C is switching off: Vc = -524 V
```

```
c Phase A is switching on:  Va = +524 V


c Calculate the initial torque value
      xpar(1) = cura
      xpar(2) = curb
      xpar(3) = curc
      xpar(4) = oldposition
      newposition = oldposition
      CALL eesolv(xpar,ypar)
      torq1= ypar(4)
      Tm = ypar(4)

c determining the phase voltages

      do k=1,N+1
            va(k)=524.0d0
            vc(k)=-524.0d0
      end do

      write(33,131) cura,curb,curc,oldposition,torq1,oldspeed+delspeed


c start with the loop

      do k =2,N+1
       print *,'Step = ',k
       xpar(1) = cura
       xpar(2) = curb

c first change the current of phase C:

       xpar(3)=curc+deli
      xpar(4)=oldposition
      if(curc.gt.0.5d0)then
        CALL eesolv(xpar,ypar)
      end if
       fiaa = ypar(1)
       fibb = ypar(2)
      ficc = ypar(3)


c again
      xpar(3)=curc-deli
      if(curc.gt.0.5d0)then
        CALL eesolv(xpar,ypar)
      end if
      fiaaa = ypar(1)
      fibbb = ypar(2)
      ficcc = ypar(3)
       ducc = (ficc-ficcc)/(2.0d0*deli)
       duac = (fiaa-fiaaa)/(2.0d0*deli)
```

```
c secondly change the current of phase A:

      xpar(1)=cura+deli
      if(curc.le.0.5d0)then
        curc = 0.0d0
      end if
      xpar(3) = curc
      CALL eesolv(xpar,ypar)
      fiaa = ypar(1)
      fibb = ypar(2)
      ficc = ypar(3)

c again

      xpar(1) = cura-deli
      CALL eesolv(xpar,ypar)
      fiaaa = ypar(1)
      fibbb = ypar(2)
      ficcc = ypar(3)
      duaa =(fiaa-fiaaa)/(2.0d0*deli)
      duca =(ficc-ficcc)/(2.0d0*deli)

c third step change the position:

      xpar(1) = cura
      xpar(4) = oldposition+delthet
      CALL eesolv(xpar,ypar)
      fiaa = ypar(1)
      fibb = ypar(2)
      ficc = ypar(3)

c again

      xpar(4) = oldposition-delthet
      CALL eesolv(xpar,ypar)
      fiaaa = ypar(1)
      fibbb = ypar(2)
      ficcc = ypar(3)
      consa= (fiaa-fiaaa)/(delthet*PI/90.0d0)
      consc= (ficc-ficcc)/(delthet*PI/90.0d0)

c We finished determining the paramaters:
c Next step is to determine new currents:
c Afterwards determine the new speed:
c Then determine the new position:
c Finally,determine the new torque:

c Determining the new currents
```

```
      if(curc.gt.0.5d0)then
        a1=(va(k-1)*del_t-cura*r_s*del_t+duaa*cura+duac*curc-
     &  consa*del_t*oldspeed)
        a2=(vc(k-1)*del_t-curc*r_s*del_t+ducc*curc+duca*cura-
     &  consc*del_t*oldspeed)
        anew=(a1*ducc-a2*duac)/(duaa*ducc-duca*duac)
        cnew=(a2*duaa-a1*duca)/(duaa*ducc-duca*duac)
        cura = anew
           curc = cnew
      else if(curc.le.0.5d0)then
        a1=(va(k-1)*del_t-cura*r_s*del_t+duaa*cura-
     &  consa*del_t*oldspeed)
        anew=a1/duaa
              cnew = 0.0d0
        cura = anew
        curc = cnew
      end if

c determine the new speed with the knowledge of the old torque:

       delspeed = (1/J)*(Tm - Tld)*del_t
       newspeed = oldspeed+delspeed
      print*,'newspeed=',newspeed
       oldspeed = newspeed

c Determine the new position:


        delposition = newspeed*del_t
       newposition = delposition*(180.0d0/PI)+oldposition
       print*,'newposition=',newposition
       oldposition=newposition

c Determine the new torque:
       xpar(1)=cura
       xpar(3)=curc
       xpar(4)=oldposition
       CALL eesolv(xpar,ypar)
       Tm = ypar(4)

       if (newposition.ge.30.0d0) then
         go to 33
       end if


       print*,'anew,cnew,newposition = ',anew,cnew,newposition,newposition
      write(33,131) anew,curb,cnew,newposition,Tm,newspeed
      write(44,100) duaa*1000.0d0,ducc*1000.0d0,
     &  duac*1000.0d0,duca*1000.0d0,newposition

       end do
```

```
c Getting out of the loop


  33 continue


c THE NEXT TWO PHASES (a & b) ARE OPERATING
c ***************************************************************************
c Case 2: Phases B and A are active: 30-60 degrees
c Phase A is switching off : Va = -500V
c Phase B is switching on : Vb = 500V

c Case 2 initialization:

c initialize

     curc = 0.0d0
     curb = 0.0d0
     cura = 82.555d0
     oldspeed = 164.774
     oldposition = 30.0d0
     newposition = oldposition


c Calculate the initial torque value

     xpar(1) = cura
     xpar(2) = curb
     xpar(3) = curc
     xpar(4) = oldposition
     newposition = oldposition
     CALL eesolv(xpar,ypar)
     torq1= ypar(4)
     Tm = ypar(4)

c determining the phase voltages

     do k=31,N+1
            vb(k)=524.0d0
            va(k)=-524.0d0
     end do


     write(33,131) cura,curb,curc,oldposition,torq1,oldspeed+delspeed

c start with the loop

     do k =32,N+1
      print *,'Step = ',k
      xpar(2) = curb
```

```
      xpar(3) = curc

c first change the current of phase A:

      xpar(1)=cura+deli

      xpar(4)=oldposition
      if(cura.gt.0.5d0)then
        CALL eesolv(xpar,ypar)
      end if
       fiaa = ypar(1)
       fibb = ypar(2)
       ficc = ypar(3)

c again

      xpar(1)=cura-deli
      if(cura.gt.0.5d0)then
        CALL eesolv(xpar,ypar)
      end if
      fiaaa = ypar(1)
      fibbb = ypar(2)
      ficcc = ypar(3)
       duaa = (fiaa-fiaaa)/(2.0d0*deli)
       duba = (fibb-fibbb)/(2.0d0*deli)

c secondly change the current of phase B:

      xpar(2)=curb+deli
      if(cura.le.0.5d0)then
        cura = 0.0d0
      end if
      xpar(1) = cura
      CALL eesolv(xpar,ypar)
      fiaa = ypar(1)
      fibb = ypar(2)
      ficc = ypar(3)

c again

      xpar(2) = curb-deli
      CALL eesolv(xpar,ypar)
      fiaaa = ypar(1)
      fibbb = ypar(2)
      ficcc = ypar(3)
      dubb =(fibb-fibbb)/(2.0d0*deli)
      duab =(fiaa-fiaaa)/(2.0d0*deli)

c third step change the position:
```

```
      xpar(2) = curb
      xpar(4) = oldposition+delthet
      CALL eesolv(xpar,ypar)
      fiaa = ypar(1)
      fibb = ypar(2)
      ficc = ypar(3)

c again

      xpar(4) = oldposition-delthet
      CALL eesolv(xpar,ypar)
      fiaaa = ypar(1)
      fibbb = ypar(2)
      ficcc = ypar(3)
      consb = (fibb-fibbb)/(delthet*PI/90.0d0)
      consa = (fiaa-fiaaa)/(delthet*PI/90.0d0)

c We finished determining the paramaters:
c Next step is to determine new currents:
c Afterwards determine the new speed:
c Then determine the new position:
c Finally,determine the new torque:

c Determining the new currents

     if(cura.gt.0.5d0)then
       a1=(vb(k-1)*del_t-curb*r_s*del_t+dubb*curb+duba*cura-
    &  consb*del_t*oldspeed)
       a2=(va(k-1)*del_t-cura*r_s*del_t+duaa*cura+duab*curb-
    &  consa*del_t*oldspeed)
       bnew=(a1*duaa-a2*duba)/(dubb*duaa-duab*duba)
       anew=(a2*dubb-a1*duab)/(dubb*duaa-duab*duba)
       curb = bnew
       cura = anew
     else if(cura.le.0.5d0)then
       a1=(vb(k-1)*del_t-curb*r_s*del_t+dubb*curb-
    &  consb*del_t*oldspeed)
       bnew=a1/dubb
            anew = 0.0d0
       curb = bnew
       cura = anew
      end if

c determine the new speed with the knowledge of the old torque:

        delspeed = (1/J)*(Tm - Tld)*del_t
      newspeed = oldspeed+delspeed
      oldspeed = newspeed
```

```
c Determine the new position:

      delposition = newspeed*del_t
      print*,'delposition=',delposition
      print*,'oldposition=',oldposition
      newposition = delposition*(180.0d0/PI)+oldposition
      oldposition=newposition


c Determine the new torque:

      xpar(2)=curb
      xpar(1)=cura
      xpar(4)=oldposition
      CALL eesolv(xpar,ypar)
      Tm = ypar(4)
        print*,'Tm = ',Tm


      if (newposition.ge.60.0d0) then
        go to 32
      end if

     print*,'bnew,anew,newposition = ',bnew,anew,newposition,newposition
     write(33,131) anew,bnew,curc,newposition,Tm,newspeed
     write(44,100) dubb*1000.0d0,duaa*1000.0d0,
    &  duba*1000.0d0,duab*1000.0d0,newposition

      end do

c Getting out of the loop

  32 continue

c THE NEXT TWO PHASES (b & c) ARE OPERATING
C
**********************************************************************

c Case 3: Phases C and B are active: 60-90 degrees
c Phase A is switching off : Vb = -500V
c Phase B is switching on : Vc = 500V

c Case 2 initialization:

c initialize

      curc = 0.0d0
      cura = 0.0d0
      curb = 83.332d0
      oldspeed = 164.959d0
```

```
      oldposition = 60.0d0
      newposition = oldposition

c start with the loop c Calculate the initial torque value

      xpar(2) = curb
      xpar(3) = curc
      xpar(1) = cura
      xpar(4) = oldposition
      CALL eesolv(xpar,ypar)
      torq1= ypar(4)
      Tm = ypar(4)

c determining the phase voltages

      do k=61,N+1
            vc(k)=524.0d0
            vb(k)=-524.0d0
      end do

       write(33,131) cura,curb,curc,oldposition,torq1,oldspeed+delspeed




      do k =62,N+1
       print *,'Step = ',k
       xpar(3) = curc
       xpar(1) = cura

c first change the current of phase B:

       xpar(2)=curb+deli
       xpar(4)=thet(k-1)
      xpar(4)=newposition
      if(curb.gt.0.5d0)then
        CALL eesolv(xpar,ypar)
      end if
       fiaa = ypar(1)
       fibb = ypar(2)
       ficc = ypar(3)

c again
      xpar(2)=curb-deli
      if(curb.gt.0.5d0)then
       CALL eesolv(xpar,ypar)
      end if
      fiaaa = ypar(1)
      fibbb = ypar(2)
```

```
      ficcc = ypar(3)
       dubb = (fibb-fibbb)/(2.0d0*deli)
       ducb = (ficc-ficcc)/(2.0d0*deli)

c secondly change the current of phase C:

      xpar(3)=curc+deli
      if(curb.le.0.5d0)then
        curb = 0.0d0
      end if
      xpar(2) = curb
      CALL eesolv(xpar,ypar)
      fiaa = ypar(1)
      fibb = ypar(2)
      ficc = ypar(3)

c again

      xpar(3) = curc-deli
      CALL eesolv(xpar,ypar)
      fiaaa = ypar(1)
      fibbb = ypar(2)
      ficcc = ypar(3)
      ducc =(ficc-ficcc)/(2.0d0*deli)
      dubc =(fibb-fibbb)/(2.0d0*deli)

c third step change the position:

      xpar(3) = curc
      xpar(4) = oldposition+delthet
      CALL eesolv(xpar,ypar)
      fiaa = ypar(1)
      fibb = ypar(2)
      ficc = ypar(3)

c again

      xpar(4) = oldposition-delthet
      CALL eesolv(xpar,ypar)
      fiaaa = ypar(1)
        fibbb = ypar(2)
      ficcc = ypar(3)
      consc= (ficc-ficcc)/(delthet*PI/90.0d0)
      consb= (fibb-fibbb)/(delthet*PI/90.0d0)

c We finished determining the paramaters:
c Next step is to determine new currents:
c Afterwards determine the new speed:
c Then determine the new position:
```

```
c Finally,determine the new torque:

c Determining the new currents

      if(curb.gt.0.5d0)then
        a1=(vc(k-1)*del_t-curc*r_s*del_t+ducc*curc+ducb*curb-
     &  consc*del_t*oldspeed)
        a2=(vb(k-1)*del_t-curb*r_s*del_t+dubb*curb+dubc*curc-
     &  consb*del_t*oldspeed)
        cnew=(a1*dubb-a2*ducb)/(ducc*dubb-dubc*ducb)
        bnew=(a2*ducc-a1*dubc)/(ducc*dubb-dubc*ducb)
        curc = cnew
        curb = bnew
      else if(curb.le.0.5d0)then
        a1=(vc(k-1)*del_t-curc*r_s*del_t+ducc*curc-
     &  consc*del_t*oldspeed)
        cnew=a1/ducc
              bnew = 0.0d0
        curc = cnew
        curb = bnew
      end if

c determine the new speed with the knowledge of the old torque:

      delspeed = (1/J)*(Tm - Tld)*del_t
      newspeed = oldspeed+delspeed
      oldspeed = newspeed

c Determine the new position:

      delposition = newspeed*del_t
      newposition = delposition*(180.0d0/PI)+oldposition
      oldposition=newposition


c Determine the new torque:
      xpar(3)=curc
      xpar(2)=curb
      xpar(4)=oldposition
      CALL eesolv(xpar,ypar)
      Tm = ypar(4)

      if (newposition.ge.91.0d0) then
        stop
      end if

      print*,'cnew,bnew,newposition = ',cnew,bnew,newposition
      write(33,131) cura,bnew,cnew,newposition,Tm,newspeed
      write(44,100) ducc*1000.0d0,dubb*1000.0d0,
     &  ducb*1000.0d0,dubc*1000.0d0,newposition
```

```
      end do

c Getting out of the loop


      close (33,status='keep')
      close (44,status='keep')
      end

c end of simulation program

c ******************************************************
```

Appendix C-Control Program for the DSP controller on the Single Pulse Control of the SR motor

```
/*****************************************************************/
/*          SRM Project. */
/*    (Control program for single pulse control on a SRM)   */
/* This program is the control program for the DSP controller  */
/* which implements single pulse control. From the reference   */
/* and position the state is determant and the acording phase  */
/* is switched on. */
/* The switching frequency is 5kHz.      */
/*****************************************************************/

#include "F240.h"

volatile unsigned int *EVIMRA  = (volatile unsigned int *) 0x742C;
volatile unsigned int *ACTR    = (volatile unsigned int *) 0x7413;
volatile unsigned int *SACTR   = (volatile unsigned int *) 0x7414;
volatile unsigned int *DBTCON          = (volatile unsigned int *) 0x7415;
volatile unsigned int *COMPR1  = (volatile unsigned int *) 0x7417;
volatile unsigned int *COMPR2  = (volatile unsigned int *) 0x7418;
volatile unsigned int *COMPR3  = (volatile unsigned int *) 0x7419;
volatile unsigned int *SCOMPR1  = (volatile unsigned int *) 0x741A;
volatile unsigned int *SCOMPR2  = (volatile unsigned int *) 0x741B;
volatile unsigned int *SCOMPR3  = (volatile unsigned int *) 0x741C;
volatile unsigned int *COMCON          = (volatile unsigned int *) 0x7411;
volatile unsigned int *GPTCON  = (volatile unsigned int *) 0x7400;
volatile unsigned int *T1CON   = (volatile unsigned int *) 0x7404;
volatile unsigned int *Datain  = (volatile unsigned int *) 0x8000;
volatile unsigned int *Datauit = (volatile unsigned int *) 0x8001;

int         Ref, Temp, rotorpos;

/******************/
/*  Main Program  */
/******************/

void main(void)
{
            /* Initialise the interupt routines */

        *EVIMRA |= 0x0001;/* Enable the PDPINT interupt routine */

            /* Initialise the PWM with no Dead time routine */

        *ACTR  = 0x0999;   /* Set PWM1, PWM3 and PWM5 to active low */
                           /* Set PWM2, PWM4 and PWM6 to active high */
        *SACTR  = 0x0015;  /* Set PWM7, PWM8 and PWM9 to active low */
        *DBTCON = 0x0000;          /* No dead time is incorporated */
        *COMCON = 0x0307;          /* Inisialize Compare PWM mode */
        *COMCON = 0x8307;          /* Inisialize Compare PWM mode */

            /* Initialize the counters */
```

```
*T1PR  = 2000;                  /* Period = 2*(50ns x 2000) = 200us */
*T1CNT = 0;                     /* Initial value of counter = 0 */
*T1CON = 0x0A802; /* Initialize counter #1 */
*T1CON = 0x0A842; /* Initialize counter #1 */

        /* Initialize the ADs */

*OCRA |= 0x070F;                /* Make ADC0, ADC1, ADC8 and ADC9 active */
ADCTRL2 -> ADCPSCALE   = 0;     /* Prescale value = 0 */
ADCTRL2 -> ADCEVSOC    = 0;
ADCTRL2 -> ADCEXTSOC   = 0;
ADCTRL1 -> suspend_free = 0;
ADCTRL1 -> suspend_soft = 1;
ADCTRL1 -> ADCIMSTART  = 0;
ADCTRL1 -> ADCINTEN    = 0;     /* Disable interrupt */
ADCTRL1 -> ADCCONRUN   = 0;     /* Single conversion */
ADCTRL1 -> ADC2EN      = 1; /* Enable ADC2 */
ADCTRL1 -> ADC1EN      = 1; /* Enable ADC1 */

        /* Resolver initialization */

*SYSCR &= 0xFF3F;/* Make IOPC1 a IO Pin */
*PCDATDIR = 0x0303;             /* Make IOPC1 (FREEZE) = 1 */
                                /* Make IOPC0 (ENABLE) = 1 */

        /* Set the COMPR registers to 0% duty cycle */

*COMPR1  = 0;
*COMPR2  = 0;
*COMPR3  = 0;
*SCOMPR1 = 0;
*SCOMPR2 = 0;
*SCOMPR3 = 0;

        /* Initialise the DACs */

*PBDATDIR |= 0x7878;            /* Make IOPB3..6 high and output signals */

do
        {
        } while (!(*T1CNT == 0));

for(;;) /* Start of endless loop for control program */
        {
                /* Read data in from ADC0 (Is reference) */

        ADCTRL1 -> ADC1CHSEL = 0;               /* Select ADC5 to read */
        ADCTRL1 -> ADC2CHSEL = 1;               /* Select ADC10 to read */
        ADCTRL1 -> ADCSOC = 1;                  /* Start conversion */
```

```
while (ADCTRL1 -> ADCINTFLAG == 0); /* Wait for conversion of data */
ADCTRL1 -> ADCINTFLAG = 1;                    /* Reset vlag */
Ref  = *ADCFIFO1;                    /* Read data in */
Temp = *ADCFIFO2;
Ref = Ref >> 6;                               /* Make Ref a 16 bit value */
Ref &= 0x03FF;
Ref = Ref * 2;

        /* Read the position of the rotor */

*PCDATDIR &= 0xFFFD;

/* Make IOPC1 (FREEZE) = 0 */
*PCDATDIR &= 0xFFFE;                          /* Make IOPC0 (ENABLE) = 0 */
rotorpos = *Datain;
*PCDATDIR = 0x0303;                           /* Make IOPC1 (FREEZE) = 1 */

                /* Make IOPC0 (ENABLE) = 1 */
rotorpos = rotorpos >> 4;            /* Make position a 16 bit value */
rotorpos &= 0x0FFF;
 rotorpos = rotorpos + 3567;         /* Zero of angle */
 if (rotorpos > 4095)
        {rotorpos = rotorpos - 4095;}

        /* Select switching state */

if (rotorpos > 0 && rotorpos < 341)    /* 0 - 30 */
{       *SCOMPR1 = 0;
        *COMPR1  = 0;
        *SCOMPR2 = 0;
        *COMPR2  = 0;
        *SCOMPR3 = 3000;
        *COMPR3  = Ref;}

if (rotorpos > 341 && rotorpos < 682)  /* 30 - 60 */
{       *COMPR1  = 0;
        *SCOMPR1 = 0;
        *COMPR2  = Ref;
        *SCOMPR2 = 3000;
        *COMPR3  = 0;
        *SCOMPR3 = 0;}

if (rotorpos > 682 && rotorpos < 1023) /* 60 - 90 */
{       *COMPR1  = Ref;
        *SCOMPR1 = 3000;
        *SCOMPR2 = 0;
        *COMPR2  = 0;
        *COMPR3  = 0;
        *SCOMPR3 = 0;}

if (rotorpos > 1023 && rotorpos < 1365) /*90 - 120*/
{       *SCOMPR1 = 0;
        *COMPR1  = 0;
        *SCOMPR2 = 0;
        *COMPR2  = 0;
        *SCOMPR3 = 3000;
        *COMPR3  = Ref;}
```

```
if (rotorpos > 1365 && rotorpos < 1706) /* 120 - 150 */
{       *COMPR1  = 0;
        *SCOMPR1 = 0;
        *COMPR2  = Ref;
        *SCOMPR2 = 3000;
        *COMPR3  = 0;
        *SCOMPR3 = 0;}

if (rotorpos > 1706 && rotorpos < 2047) /* 150 - 180 */
{       *COMPR1  = Ref;
        *SCOMPR1 = 3000;
        *SCOMPR2 = 0;
        *COMPR2  = 0;
        *COMPR3  = 0;
        *SCOMPR3 = 0;}

if (rotorpos > 2047 && rotorpos < 2388)  /* 180 - 210 */
{       *SCOMPR1 = 0;
        *COMPR1  = 0;
        *SCOMPR2 = 0;
        *COMPR2  = 0;
        *SCOMPR3 = 3000;
        *COMPR3  = Ref;}

if (rotorpos > 2388 && rotorpos < 2730)  /* 210 -240 */
{       *COMPR1  = 0;
        *SCOMPR1 = 0;
        *COMPR2  = Ref;
        *SCOMPR2 = 3000;
        *COMPR3  = 0;
        *SCOMPR3 = 0;}

if (rotorpos > 2730 && rotorpos < 3071) /* 240 - 270 */
{       *COMPR1  = Ref;
        *SCOMPR1 = 3000;
        *SCOMPR2 = 0;
        *COMPR2  = 0;
        *COMPR3  = 0;
        *SCOMPR3 = 0;}

if (rotorpos > 3071 && rotorpos < 3412) /*270 - 300*/
{       *SCOMPR1 = 0;
        *COMPR1  = 0;
        *SCOMPR2 = 0;
        *COMPR2  = 0;
        *SCOMPR3 = 3000;
        *COMPR3  = Ref;}

if (rotorpos > 3412 && rotorpos < 3753) /* 300 - 330 */
```

```
{       *COMPR1  = 0;
        *SCOMPR1 = 0;
        *COMPR2  = Ref;
        *SCOMPR2 = 3000;
        *COMPR3  = 0;
        *SCOMPR3 = 0;}

if (rotorpos > 3753 && rotorpos < 4095) /* 150 - 180 */
{       *COMPR1  = Ref;
        *SCOMPR1 = 3000;
        *SCOMPR2 = 0;
        *COMPR2  = 0;
        *COMPR3  = 0;
        *SCOMPR3 = 0;}

        /* Write to the DACs */

*PBDATDIR &= 0xFFF7;                    /* Select DAC 1 */
*Datauit  = Ref;                /* Write Reference to DAC */
*PBDATDIR |= 0x0008;

*PBDATDIR &= 0xFFEF;                    /* Select DAC 2 */
*Datauit  = *COMPR1;                    /* Write Ia(Reference) to DAC */
*PBDATDIR |= 0x0010;

*PBDATDIR &= 0xFFBF;                    /* Select DAC 4 */
*Datauit  = rotorpos;           /* Write Mechanical rotor angle to DAC */
*PBDATDIR |= 0x0040;

do
        {
        } while (*T1CNT > 5);

}
}
```

Bibliography

[1] R. Krishnan, Switched Reluctance Motor Drives. "Modeling, Simulation, Analysis, Design, and Applications"

[2] J Robert, D.K., "Modelling and Simulation of Electrical Machines and Power Systems", Transaction (Editors) Elsevier Science Publishers B.V. (North Holland) © Imacs, 1988

[3] R. Arumugam, D.A. Lowther, R. Krishnan, J.F. Lindsay. "Magnetic Field Analysis of a Switched Reluctance Motor using a Two Dimensional Finite Element Model".

[4] M.J. Kamper, "Design Optimisation Of Cageless Flux Barrier Rotor Reluctance Synchronous Machine". PHD Dissertation, University of Stellenbosch 1996

[5] "Variable-Speed Switched Reluctance Motors". IEEE Proc., Vol.127, Pt. B, No.4, July 1980

[6] L.M.M. Sitsha , "Design Of Tapered And Straight Stator Pole Switched Reluctance Machine". MSc.Eng. Thesis, University of Stellenbosch 2000

[7] Yinghui Lu, "Instantaneous Torque Control Of The Switched Reluctance Motor". MSc. Eng. Thesis, University of Tennessee, Knoxville, August 2002

[8] Corda, J and Masica, S "Computation of Torque Pulsations Of The Switched Reluctance Motor Drive". IEEE Conference Publication, No.310, 1989, pp 308-311.

[9] Stephenson, J.M. and Corda, J "Computation of Torque and Current in Double Salient Reluctance Motors from Non-Linear Magnetisation Data", IEEE Proc., Vol.122, No.5, 1979, pp. 393-396.

[10] Pieter D. Fick " Evaluation Of The Constant Current Angle Controlled Reluctance Synchronous Machine Drive", MSc.Eng. Thesis, University Of Stellenbosch 2002

[11] William F. Ray, Peter J. Lawrenson "High-Performance Switched Reluctance Brushless Drives, IEEE Transactions On industry Applications. Vol. 1A-22.No.4 July/August 1986

[12] T. J. E. Miller, Brushless Permanent-Magnet and Reluctance Motor Drives, Oxford Science Publications, 1989

[13] J.W. Van Der Merwe, "Design of a 50 kW power electronic converter for a switched reluctance machine", final year project, University of Stellenbosch 2000

[14] S. Vukosavic et al., "SRM inverter topologies: A comparative evaluation" IEEE Trans. Ind. Applicat., vol. 27, pp 1034-1047, Nov./Dec. 1991.

[15] Frede Blaabjerg, Phillip C. Peter Omand Rasmussen, and Calum Cossar, " Improved Digital Current Control Methods in Switched Reluctance Motor Drives". IEEE Transactions on Power Electronics, Vol. 14, no. 3, May 1999

[16] Roger C. Becerra, Mehrdad Ehsani, and J.E. Miller. " Commutation of SR motors". IEEE Transactions on Power Electronics, Vol. 8, No. 3, July 1993

[17] Philip C. Kjaer, Gabriel Gallegos-López. "Single-Sensor Current Regulation in Switched Reluctance Motor Drives". IEEE Transaction on Industry Applications, Vol. 34, No. 3 May/June 1998

[18] Arnold Ostebee and Paul Zorn "Calculus from graphical, Numerical and Symbolic Points of View". Volume 2 , Preliminary Edition, St. Olaf College 1995

[19] K. De Brabandere, J. Dreisen, and R. Belmans, "The control of Switched Reluctance Machine Drives and their Use for Flywheel Energy Storage". ESAT-ELECTA research group, K.V. LEVNEN, Kasteel park Arenberg 10, B-3001, Haverlee Belgium.